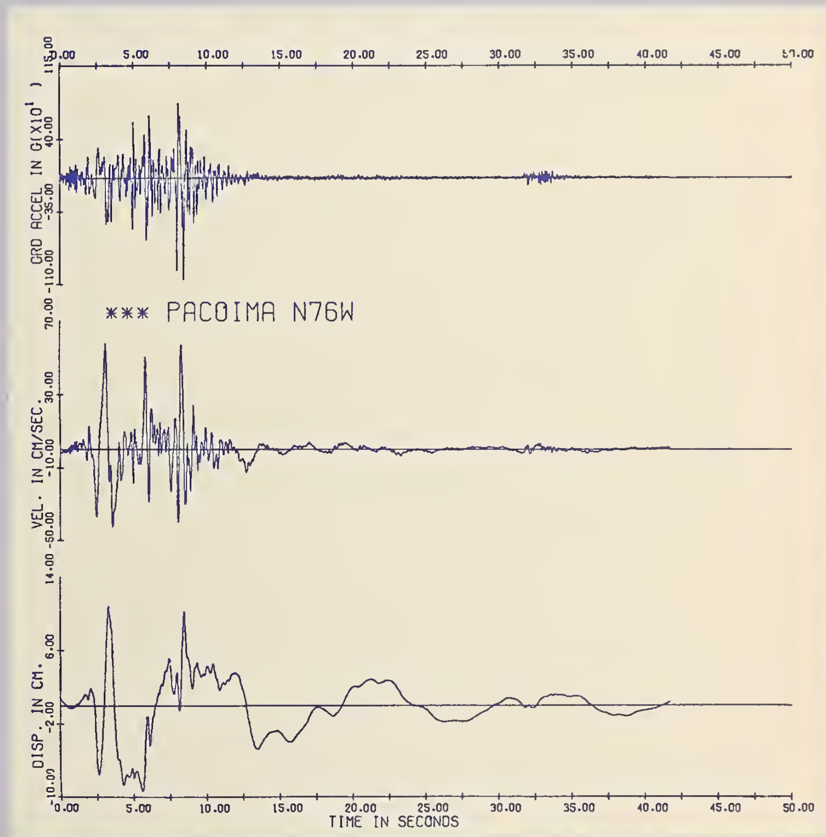
t No. FHWA-RD-77-92

TIZATION AND INTEGRATION OF STRONG MOTION ACCELEROGRAMS



DEPARTMENT OF TRANSPORTATION
UNITED STATES OF AMERICA

August 1977
Final Report

Document is available to the public through
the National Technical Information Service,
Springfield, Virginia 22161

Prepared for
FEDERAL HIGHWAY ADMINISTRATION
Offices of Research & Development
Washington, D. C. 20590

FOREWORD

This repor+ is the result of research conducted at the University of Arizona for the Federal Highway Administration (FHWA), Office of Research, under FHWA Purchase Order P.O. 5-3-0190. The report will be of interest to those researchers concerned with the earthquake analysis of highway bridges including the processing of strong motion records. It outlines procedures and operational instructions for the digitization and integration of recorded strong motion accelerograms.

Copies of the report are being distributed by FHWA transmittal memorandum. Additional copies may be obtained from the National Technical Information Service, 5285 Port Royal Road, Springfield, Virginia 22161.

Charles F. Scheffey
Charles F. Scheffey
Director, Office of Research
Federal Highway Administration

NOTICE

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
FHWA RD-77-92		

4. Title and Subtitle	
Digitization and Integration of Strong Motion Accelerograms	5. Report Date: August, 1977
	6. Performing Organization Code
7. Author(s): Haaren A. Miklofsky and William B. Mancini	8. Performing Organization Report No.
9. Performing Organization Name and Address: University of Arizona, Department of Civil Engineering, Tucson, Arizona 85721	10. Work Unit No. (TRAIS): FCP 35A2981
	11. Contract or Grant No.: FHWA P.O. 5-3-0190
12. Sponsoring Agency Name and Address: Department of Transportation, Federal Highway Administration, Office of Research, Washington, D. C. 20590	13. Type of Report and Period Covered: Final Report
	14. Sponsoring Agency Code: S0668

15. Supplementary Notes

FHWA Contract Manager: James D. Cooper (HRS-11)

16. Abstract

Procedures for the digitization and integration of Strong-Motion Accelerograms are presented. Operational instructions and digitizing procedures are described for the Electrak Data Tablet/Digitizer, the Benson-Lehner Chart Reader, and the Microfische Film Reader. Optical scanning using the Perkin-Elmer Microdensitometer is also considered; but photographic problems with some Strong-Motion Records limit the use of an optical method for digitization. The Electrak method is considered superior to other methods; the Microfische method gives excellent expedient results when commercial digitizers are unavailable.

The computer program developed by Dr. M. D. Trifunac, et.al., at the California Institute of Technology is the basis for the integration procedure. A detailed description of the program is included. It is found that long period errors introduced by the program methodology can seriously affect the accuracy of the resultant time-displacement history. Changes in the high pass filters are needed to correct for these long period errors. When long periods in the frequency range of the error frequencies are part of the strong-motion acceleration record, it may be impossible to guarantee the accuracy of the integrated time-displacement curve. The appendix of the report contains a full listing of the computer program for the CDC 6400 and IBM 360/370 computers. A User's Manual is also included.

17. Key Words	18. Distribution Statement
Accelerograms, Strong Motion, Earthquake Analysis	No restrictions. This document is available through the National Technical Information Service, Springfield, Virginia 22161.

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	295	

Form DOT F 1700.7 (8-72)

Reproduction of completed page authorized

PREFACE

The authority for conducting this investigation is the Department of Transportation, Contract No. 5-3-0190, Federal Highway Administration, Office of Research, executed with the University of Arizona, Engineering Experiment Station, College of Engineering, Tucson, Arizona 85721.

The principal investigator is Haaren A. Miklofsky, Professor of Civil Engineering and Engineering Mechanics, the University of Arizona. William B. Mancini, a graduate student in Civil Engineering, joined the principal investigator for the summer months of 1976 in the preparation of a Master's Report as part of this research.

The CDC computer research was performed at the computer center of the University of Arizona. The IBM 360/370 conversion was performed at the computer facility of the Tucson Gas and Electric Company, Tucson, Arizona. The IBM Calcomp Plotter of the Phelps Dodge Corporation at Morenci, Arizona, was used to check the plotting subroutine.

This report was prepared by Haaren A. Miklofsky and incorporates part of the Master's Report by William B. Mancini.

ACKNOWLEDGMENTS

The authors wish to thank Dr. Arthur Brady, Seismic Engineering Branch, U.S. Geological Survey, for furnishing a contact negative film copy of the Pacoima Dam Accelerogram at the beginning of this research. Thanks are also due Charles O. Meyer, Vice-President, Terra Technology for furnishing a glossy photograph of the A.R. 240 Strong-Motion Accelerograph.

Nicholas Cocavessis, Micheil Karbough, and Yiannakis Katsambos, senior students in Civil Engineering at the University of Arizona, assisted the principal investigator during preliminary investigations on instrumentation and computer programs during the spring of 1976.

The authors are indebted to Dr. Edward Shirley for assistance with the use of the Electrak Data Tablet/Digitizer at the Watershed Research Station; Tucson, Arizona, and to Victor Estrella who assisted the authors with the use of the Benson-Lehner machine at the station. The Trak 010 subroutine, Appendix E-2, was written by Steve Kuteroff, former member of the U. S. Watershed staff.

Bahram Raeen, a graduate student in Civil Engineering at the University of Arizona, redigitized the Pacoima Dam record on the Benson-Lehner machine in December 1976.

The principal investigator is indebted to Mr. Ken Saul, Vice President, Tucson Gas and Electric Company, for permission to use the Tucson Gas and Electric IBM 360/370 computer. Special thanks

are due Ken Johnson and William D. Tolbert for assistance with the JCL on that computer.

Appreciation is hereby acknowledged to Mr. John Bolles, Assistant Manager, Phelps Dodge Corporation at Morenci, Arizona, for permission to use the Calcomp Computing Facility at Phelps Dodge. Special thanks are due Ken Williams and Bob Stearley who helped the principal investigator by writing a special PL1 program for use with their computer during one long night.

Harry Goforth, Computer Center, University of Arizona, rewrote the Trak 010 subroutine for the IBM 360/370 computer to read the nine-track tapes.

The principal investigator is thankful that Dr. Manfred R. Bottaccini, Professor of Aerospace and Mechanical Engineering at the University of Arizona, became interested in the project in its last phase. Several discussions with Dr. Bottaccini helped the principal investigator pinpoint the source of errors in the Cal. Tech. integration methodology.

Finally the principal investigator owes a debt of gratitude to Jim Cooper, project manager, for his patience and understanding of the problems encountered by the authors in pursuing the research for this report.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Page

Page

Page

Page

DIGITIZATION AND INTEGRATION OF STRONG MOTION ACCELEROGRAMS

I. Introduction

1. Subject

This report is presented as a procedure for digitizing and processing strong motion earthquake accelerograms. It describes the process by which an earthquake record is used to obtain actual ground acceleration, velocity, and displacement information. The procedure is primarily for use with time-acceleration data originally recorded on film by a mechanical or optical system.

2. Background

Figure 1 shows the first 28 seconds of record at Pacoima Dam, San Fernando, California during the earthquake of February 9, 1971. A full-size contact negative of the record was furnished to the authors by Dr. A. G. Brady, of the U.S. Geological Survey. The accelerogram shown in Figure 1 was recorded by an AR-240 Strong Motion Accelerograph (Figure 2a). The AR-240 accelerograph was formerly manufactured by United ElectroDynamics Inc., and Teledyne Inc., from 1963 to approximately 1970. Its salient features are:

a. Continuous strong motion acceleration record from about 0.1 second after the initial actuating pendulum contact to 7 seconds after the last pendulum contact.

b. Storage capacity of 150-foot roll of photographic paper record for three orthogonal components of acceleration.

c. A total of eight recorded traces, comprising three fixed reference traces, three variable accelerometer traces, and two timing traces.

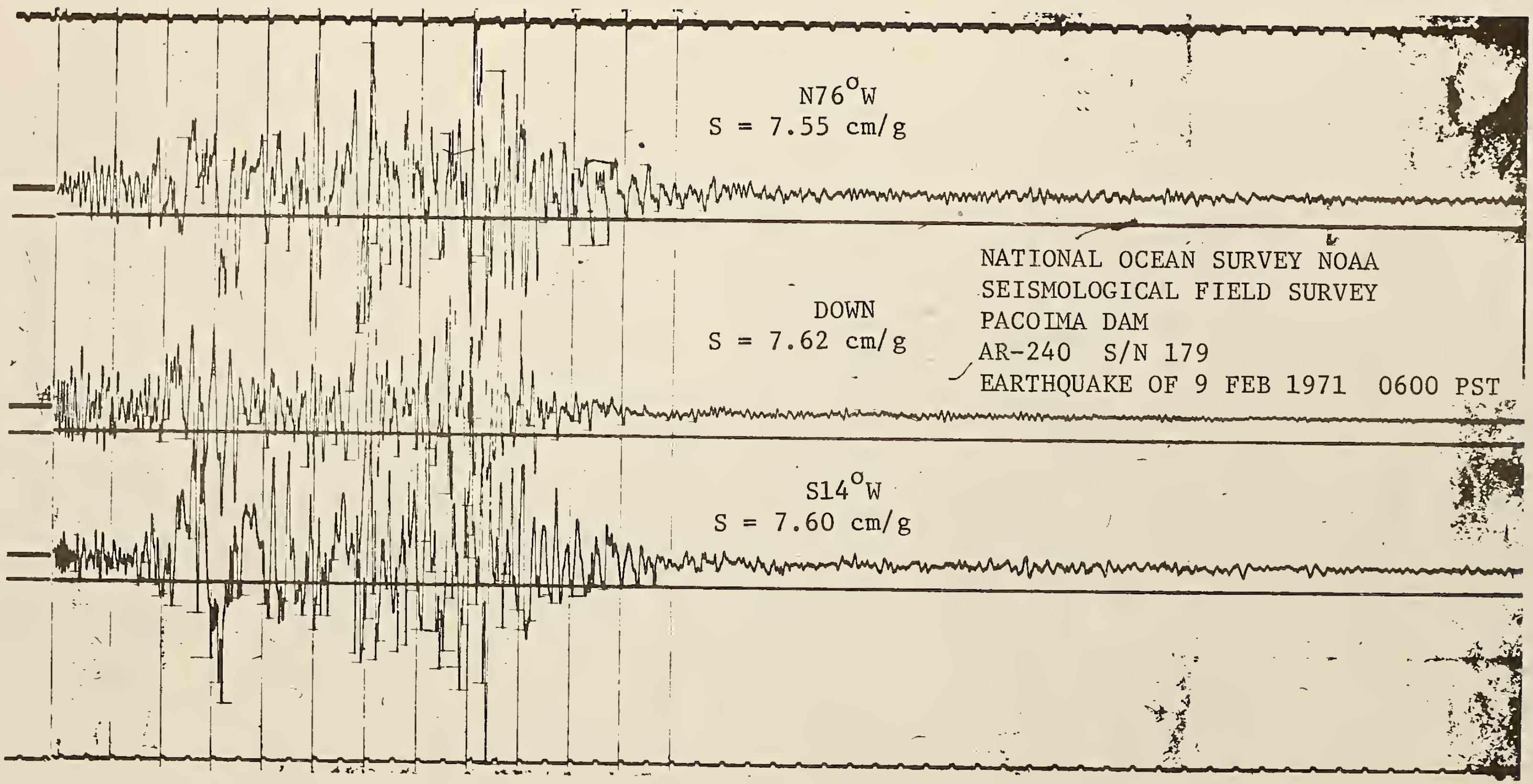


FIGURE 1. FIRST 28 SECONDS OF STRONG MOTION ACCELEROGRAM, PACOIMA DAM, SAN FERNANDO EARTHQUAKE, FEBRUARY 9, 1971
See footnote page 5. The S16°E record is identified as S14°W in other publications.

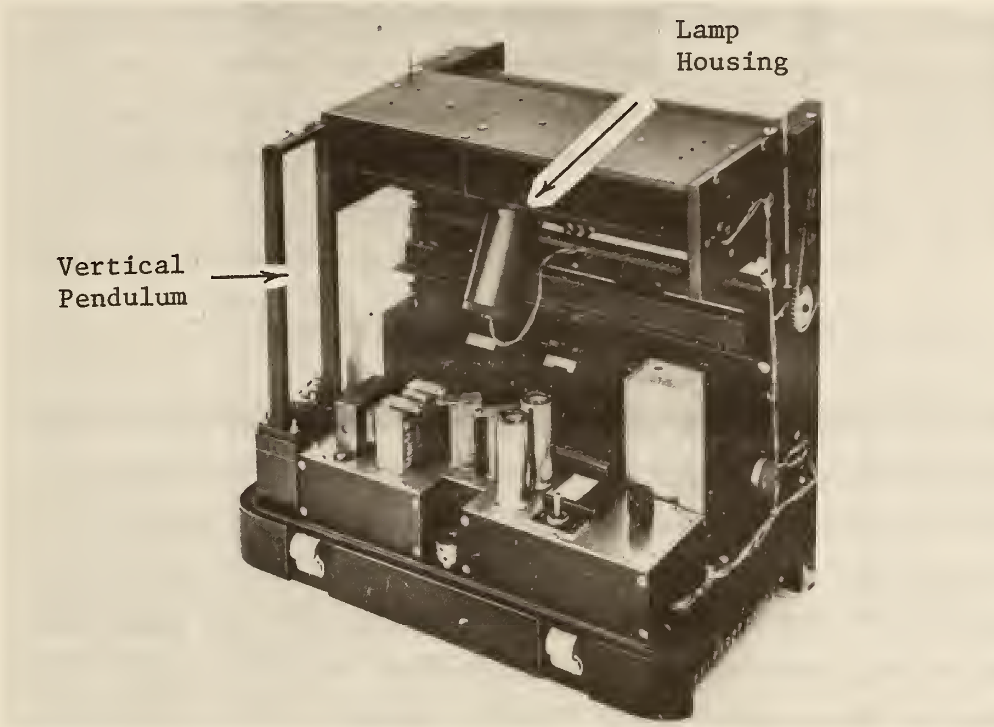


FIGURE 2-a. PHOTOGRAPH OF AR-240 STRONG MOTION ACCELEROGRAPH, COURTESY OF TERRA TECHNOLOGY CORP., SEATTLE, WASHINGTON

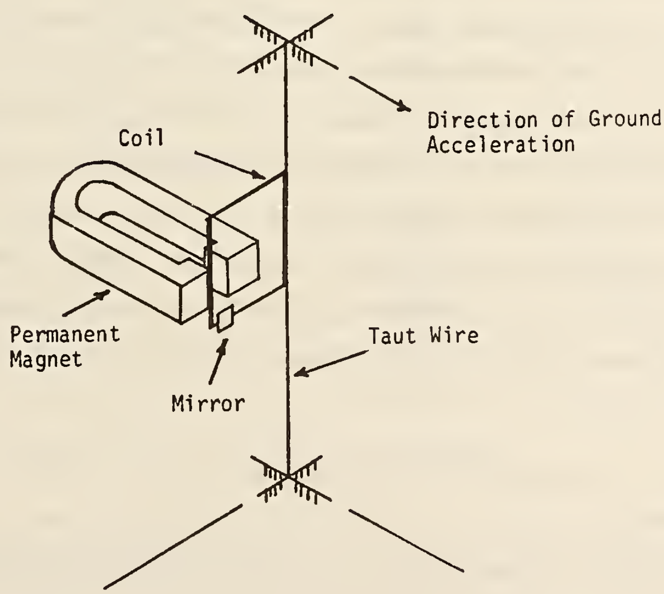


FIGURE 2-b. SCHEMATIC DIAGRAM OF TORSION SEISMOMETER

d. Light-weight (60 pounds total, not including the external batteries) and compact size (16 x 16 x 14 inches).

e. Integral calibration of seismometer period and damping.

f. Constant velocity paper speed of 2 cm/sec.

g. Adjustable sensitivity of 7.5 g and range of 0.01 to 1.0 g.

The accelerograph contains three Lehner-Griffith seismometers which are similar to the well-known Wood-Anderson torsion seismometers, using a small mirror, mounted directly on the taut suspension system and equipped with electro-magnetic damping.[1][2]* Figure 2b schematically illustrates the operation of the torsion seismometer. A seismic mass consisting of a rectangular many-turned coil of wire is excentrically attached to a taut wire. Electro-magnetic damping is accomplished by surrounding the free portion of the wire by a permanent magnet. By changing the external resistance in the coil circuit the damping can be set to any desired value. An additional advantage of the coil system is that an external electrical signal can be introduced easily into the transducer element for calibration purposes.[3] Mechanical-optical AR-240 recording is accomplished photographically by means of reflected light from a mirror attached to the mass. The AR-240 records on photographic paper 12 inches wide, and has a natural period between 0.055 and 0.065 seconds. Damping is approximately 60% of critical. Timing marks are two per second at $\pm$ 1%.

When strong earth motion occurs a mass attached to a vertical pendulum moves in a horizontal direction and closes the platinum pendulum contacts which in turn releases telephone type

---

* Numbers in ( ) refer to references listed in Appendix G.

relays controlling the drive motor, timing and control circuit, and the light source. This type of relay control coupled with a transistorized light source circuit provides a start delay timing of only about 0.1 seconds from initial contact to full operation. The minimum acceleration required to close the pendulum contacts can be less than 0.01 g.

Processing of accelerogram records takes two general steps: (1) converting the time-acceleration trace to digital information (digitization), and (2) integration and correcting. The first step is a physical task and must be performed with utmost care. The procedures used in the second step are more or less "automatic" i.e., the data is processed through a digital computer and cannot be directly influenced by the user unless he changes the computer program. One cannot emphasize the meticulous accuracy by which the digitization must be done. To put a scale on the accuracy required, let us consider Figure 3 which shows a published plot of the final acceleration, velocity, and displacement of the Pacoima Dam N76°W* component of record by Cal. Tech.[4] The original accelerogram shows a vertical scale of 7.6 centimeters per g or 7.6 centimeters per 981 centimeters/sec^2. If a vertical displacement error in digitizing points of one cm/sec^2 or approximately 0.003 inches occurred during a 30-second length of record (Figure 4), the end velocity would change by 30 cm/sec and the end displacement would change by 450 centimeters. Since the maximum displacement is only 12 centimeters, it can be seen that small errors in digitizing acceleration records can completely distort the displacement record and make it virtually impossible to correct during the integration procedure.

---

* Several publications list this as the S74°W component. However, the new component designation was printed on the record received by the authors from the U.S. Geological Survey.

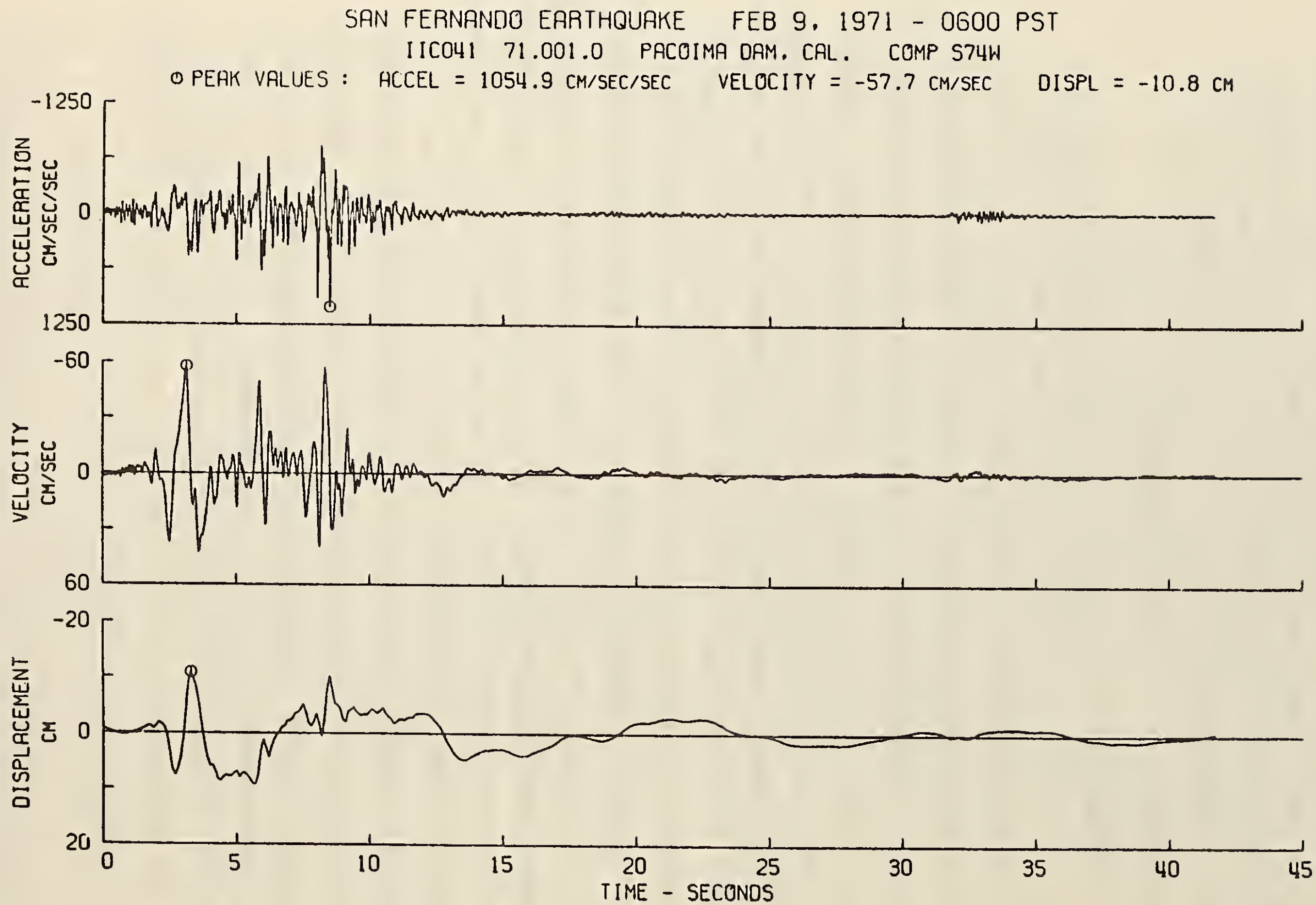


FIGURE 3. FINAL ACCELERATION, VELOCITY, AND DISPLACEMENT FOR N76°W PACOIMA DAM COMPONENT FROM CALIFORNIA INSTITUTE OF TECHNOLOGY PUBLICATION (4) See footnote pages 2 and 5.

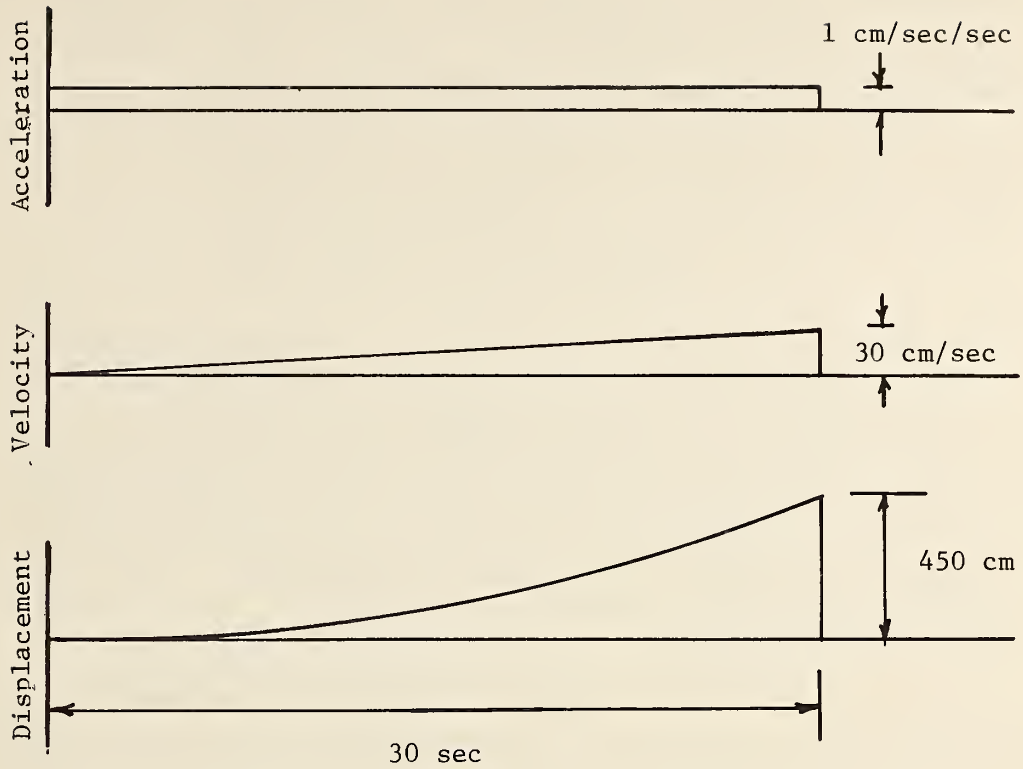


FIGURE 4. EFFECT OF A CONSTANT CM/SEC/SEC DIGITIZING ERROR ON VELOCITY AND DISPLACEMENT

3. Computer Program

Several computer programs described in the literature for the integration of earthquake records were studied,[(5),(6),(7)] but eventually it was decided to use the Cal. Tech.* program as a basis for integration so that the reader would have an existing list of digitized data of past earthquakes and Cal. Tech's so

*The computer program developed by Trifunac and Vijayaraghavan, California Institute of Technology, as given in reference 10, is referred to as the Cal. Tech. program in this report.

called "standardized integrated plots" with which to compare results for future research.[(8),(9)] The procedure outlined in this report follow those used by the California Institute of Technology, as described in reference (10) with primary differences existing mainly in the programming techniques. The program listed in reference (10) written for the IBM 360/370 machine, utilizes assembly language subroutines for reading, writing, plotting, and storage of information in a production system of records. Also the program as listed would have exceeded the storage capacity of the CDC 6400 computer at the University of Arizona. Therefore, the program was completely rewritten into a more compact form for efficient turn-around time using only Fortran IV statements for both the IBM 360/370 O.S. computer and the CDC 6400 computer.

A single program DAISMA* is used for processing of digitized input data either in card or magnetic tape form, and also for the integration of the processed data. The procedure consists of three steps:

1. Digitized data on either magnetic tape or cards are first plotted to the identical scale of the accelerogram trace using a subroutine called PLOTTR. Magnetic tape data is punched on cards. The plot of the data is overlain on the original record over a light table and the data is corrected either by hand on the keypunch machine, or by redigitizing.
2. The corrected data is next baseline corrected (subtraction of fixed trace from acceleration record) by resubmitting the card data to the program with control cards to go to subroutine PHASE1, after which baseline corrected data is punched on cards and plotted.

* DAISMA - Digitization and Integration of Strong Motion Accelerograms.

3. After the examination of the baseline corrected plot, the new cards are again resubmitted to the program with control cards to go to subroutine PHASE2, where the data is integrated to give final values of acceleration, velocity, and displacement and a plot of this information.

4. Outline of Method

The digitizing process (discussed in detail in the next section) consists of converting an analog trace (i.e., the accelerogram) to a digital record of plane coordinates. The sequence of the coordinate data points represent the accelerogram; i.e., if the individual coordinate points were plotted on a two-dimensional graph and each successive point connected by a straight line, a duplicate of the accelerogram would result. Naturally, there is a slight deviation from the original accelerogram, but these deviations become negligible as the number of digitized points increases.

There are several errors inherent in converting an analog trace to digital information. These are classified as (1) systematic errors i.e., errors which continue to occur and have the same magnitude under the same recording conditions and (2) accidental errors, i.e., errors which occur infrequently and may greatly deviate from the true value. Errors and their elimination are discussed in detail.

In order to obtain velocity and displacement from a function representing acceleration, integration of the function is necessary. The integration process is carried out by means of the "Trapezoidal Rule". This is an approximate means of integrating by numerical methods. (The method of integration is discussed in Section IV.)

A filtering technique is used in the processing of data to remove extraneous high and low frequency components of data. Thus only frequencies within a certain "band" are allowed to pass unaltered. This process is called "high (or low) pass" filtering and is discussed in Section IV.

To locate a "baseline", i.e., a line which represents zero acceleration, a least squaring procedure is executed. The data is then "fitted" to this baseline so that all quantities are given as being positive or negative relative to the baseline. Leastsquaring is necessary because the accelerogram trace does not begin until <u>after</u> the earthquake has started. (The earthquake itself triggers the accelerometer). Thus zero acceleration at the beginning of the trace is not given and must be determined by some other means. The method of leastsquaring is presented in Appendix A.

To support and more clearly demonstrate the validity of the general procedure outlined above, experiments are presented in Section VII. In these experiments acceleration traces are prepared. The traces are then processed in the same manner in which an earthquake record (accelerogram) would be processed. In the experiments, however, actual displacements are recorded on film so that they can be compared to displacements computed from the accelerograms.

To obtain similarity between integrated and measured displacements the Cal. Tech. program was slightly modified. The reasons for the modification are given in Sections VI and VII.

5. Reconstruction of Contact Negative

A 35% reduced print of a part of the contact negative is shown in Figure 5. Some of the lines of the trace were so faint that they were unobserved when the first digitizing work was

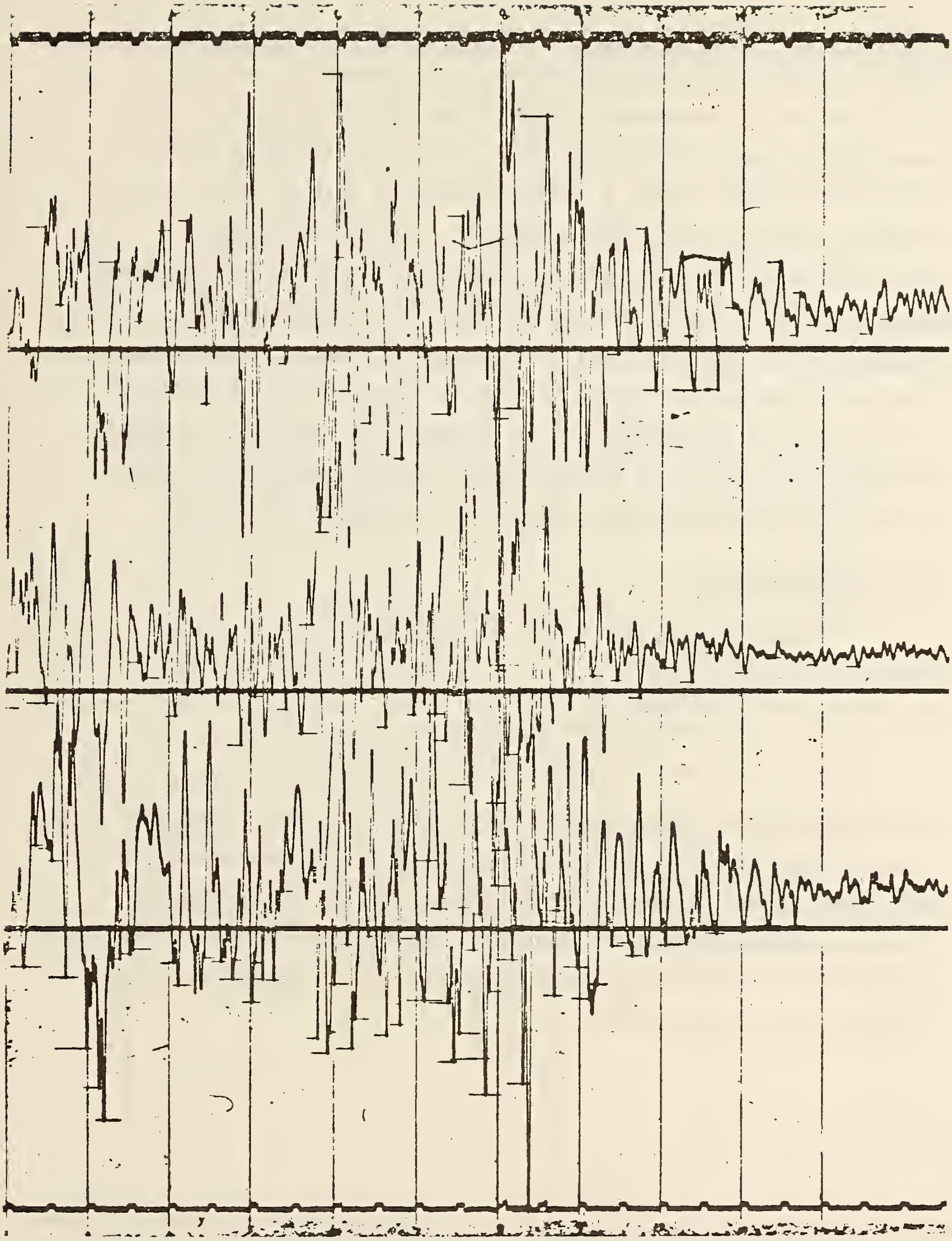


FIGURE 5. COPY OF 11 1/2 SECONDS OF PACOIMA DAM RECORD, AS ORIGINALLY RECORDED. PRINTED AT 65% OF FULL SIZE

started. Guessing at the location of these lines led to serious errors. Upon examination of the negative over a light table, the extremely faint lines were found with a magnifying glass. Thereupon they were intensified by cutting the emulsion away using a Ramsey Film Line Cutter, while still examing the area under the magnifier. At the same time, it was decided to opaque one negative copy of the record to separate the interfering curves and to blacken the background for studies by an optical digitizing system. Figure 6 shows the reconstructed record which was used in the digitizing process. During optical readout the middle curve was completely opaqued, although it could have been included and separated from top and bottom curves by reprinting from two reconstructed copies.

6. Digitizing Machines

Three digitizing machines were used.

1. The Electrak Digitizer
2. The Benson-Lehner Digitizer
3. The Perkin-Elmer Microdensitometer

The Electrak and Benson-Lehner digitizers are fully described in this report and work was accomplished to completion on these machines. Time allowed only introductory work on the Perkin-Elmer Microdensitometer; however, photographic problems (to be described) may limit the application of this machine for earthquake record digitization.

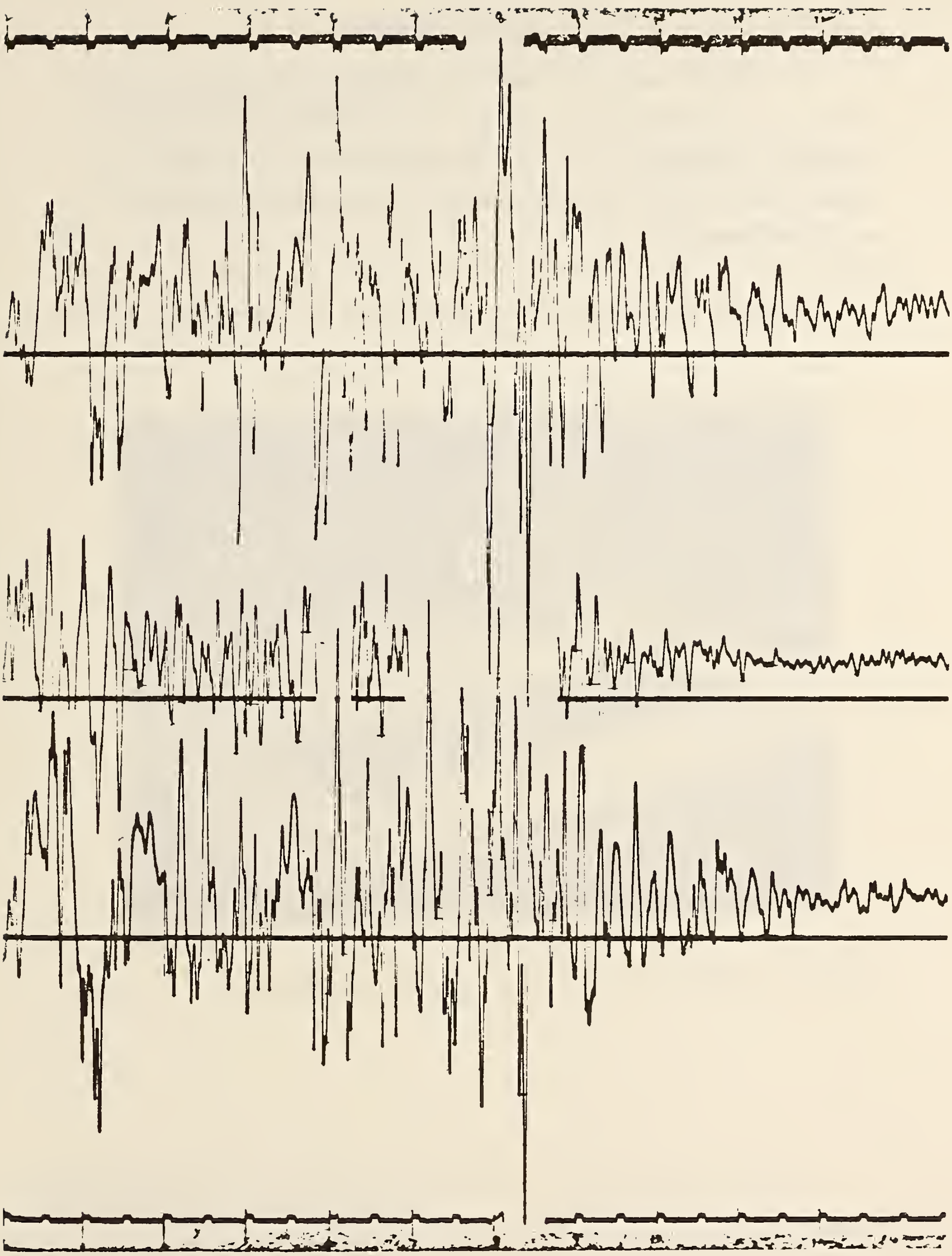


FIGURE 6. RECONSTRUCTED RECORD OF FIGURE 5 AFTER THE CURVES WERE INTENSIFIED AND THE BACKGROUND OPAQUED

7. Hand Digitizing - Microfische Film Reader

A novel method of hand digitizing a record with the aid of a microfische film reader is described in Section II. Although the method is time consuming, it is practical when commercial digitizing machines are unavailable, and gives excellent results.

II. DIGITIZATION

1. Electrak Digitizing Machine*

Figure 7 illustrates the TRAK 100** digitizer in use at the Watershed Research Station, Tucson, Arizona. The 36" x 48" active work surface is a white sheet of plastic above a gridwork of wires spaced about 1/10" apart. A current is pulsed through a single wire at the edge of the sheet of magnetic

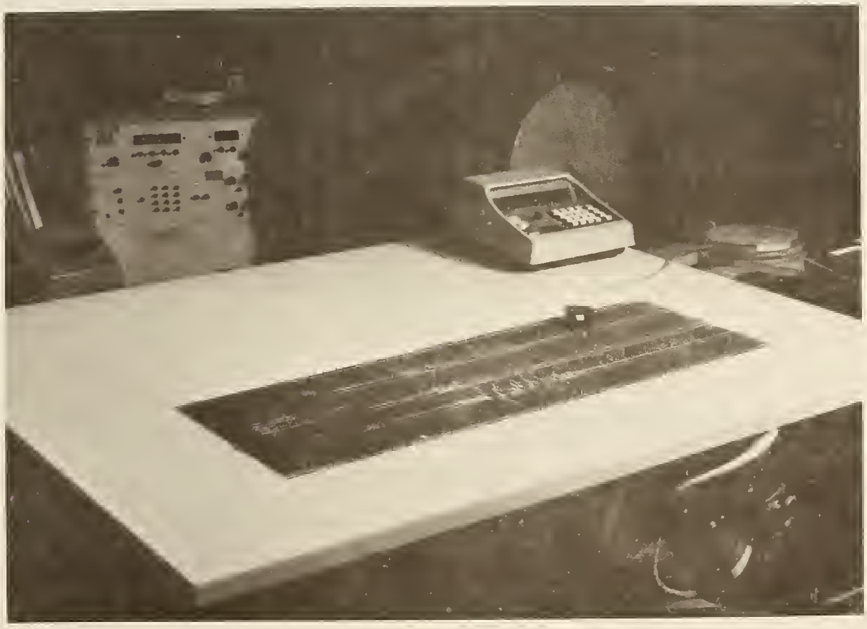


FIGURE 7. ACCELEROGRAM MOUNTED ON TABLE OF ELECTRAK DIGITIZER

* Manufactured by Electrak Corporation, 16634 Oakmont Avenue, Gaithersburg, Maryland 20760.

** The terms TRAK 100 and Electrak are used synonymously throughout this report.

material causing a planar strain wave to propagate through the sheet. On the surface of the table is a cursor which has a crosshair etched on the reading glass. As the strain wave passes beneath the cursor, an electric signal is produced in a small coil in the cursor. A digital coordinate is produced by timing the delay between the START (current) pulse and the STOP pulse. A second current carrying wire along the adjacent edge of the sheet provides the determination of the other coordinate.

Figure 8 shows the gridwork of wires located under the plastic sheet used in making a two-dimensional determination of the position of the cursor. The "Send" wire is just a one-turn coil through which a current is pulsed. Two such wires are used -- one for the X and one for the Y determinations. They are pulsed at different times to avoid ambiguity. The strain wave propagates down all the wires in one direction simultaneously. The receiving coil is in the crosshair of the cursor above the table surface. The insertion loss of the entire measuring process is low enough to permit accurate determinations even when the cursor is 3/16" above the table surface. Since position sensing is done along the wires instead of across the wires, any location errors of the wires relate to the accuracy as the cosine of the error angle. However, this is a very small number. The resolution of the table at the Watershed Research Station is 0.005" with an accuracy of $\pm$ 0.005".

Figure 9-a shows an enlargement of a section of the Pacoima Dam record. To increase the accuracy of positioning the cursor in the middle of the acceleration curve, a Bausch & Lomb measuring magnifier was attached over the cursor glass as shown in Figure 9-b, which enlarged the accelerogram curve approximately

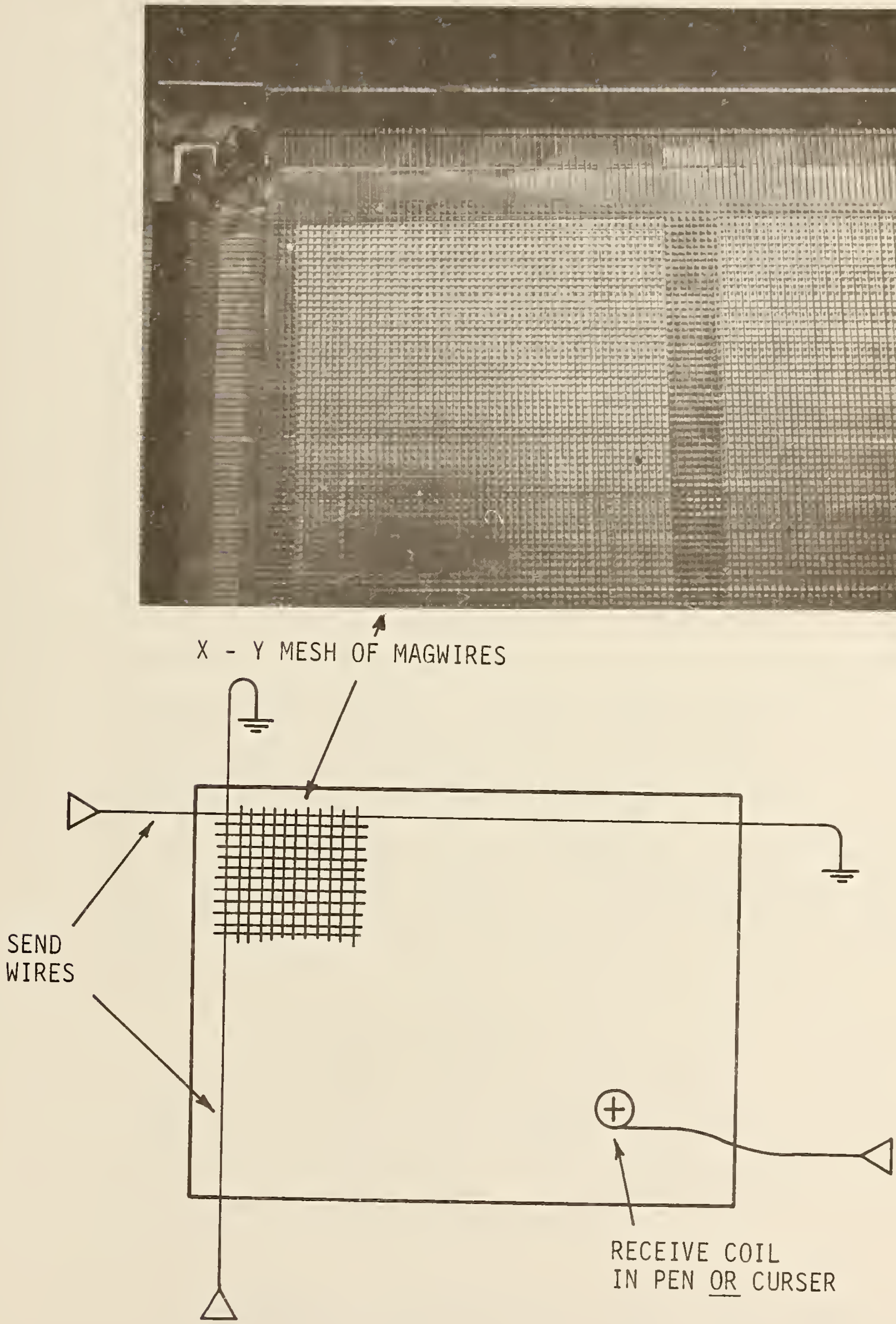
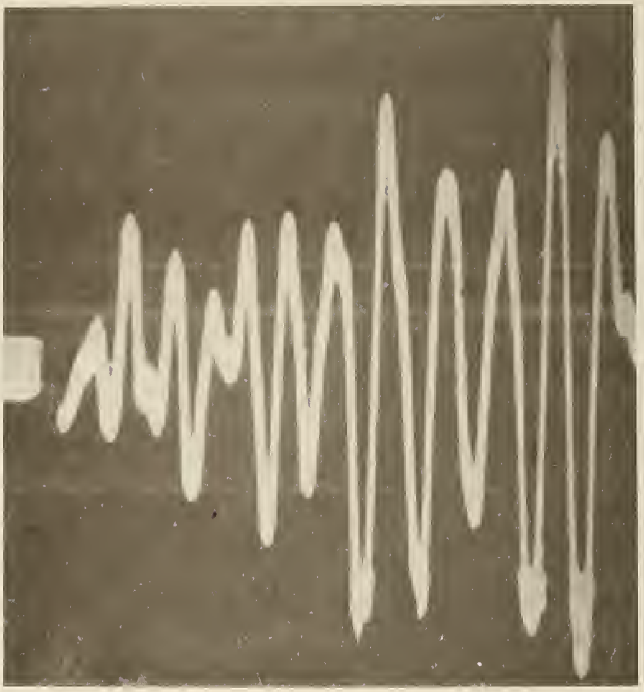
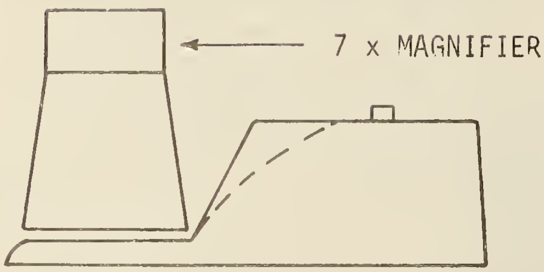


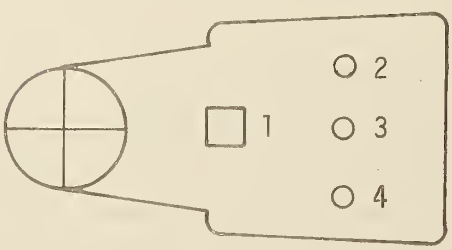
FIGURE 8. MESH OF MAGWIRES UNDER ELECTRAK PLASTIC TABLE TOP AND SCHEMATIC OF POSITION SENSING



(a)



(b)



(c)

FIGURE 9-a. MAGNIFIED PORTION OF ACCELEROGRAM

FIGURE 9-b. ELEVATION VIEW OF CURSOR

FIGURE 9-c. TOP VIEW OF CURSOR

to the width shown in Figure 9-a. The cursor (Fig. 9-c) has four buttons each of which may be pushed to record the coordinates of the point in question (located by the cross-hair). These buttons are identification buttons and will print out a four-character number along with the coordinate data and can be used to distinquish the digitized components of the accelerogram from each other. It was found convenient, however, to use the #1 button exclusively and change the curve identification on the console as will shortly be explained.

Figure 10 shows the small floating console and illustrates the position of the various controls. When the console is in operation mode, the display shows the first 16 characters of the 36 character record recorded on the tape. The first four characters, CCCC, represent a four-digit counter, whose starting number is dialed on the four decade thumb wheels shown. As each point is read into the tape this number reduces by 36 characters. When it goes to zero, it reaches the end of a block of data on the tape.

The next six characters comprise the X coordinate from an established zero reference -- with the first character representing the ± sign. Next follows six more characters representing the Y coordinate preceeded by a sign.

The next 16 characters of the record is alpha-numeric information placed in memory via the electric typewriter. When this keyboard entry is first made into memory, it will be displayed on the console when the FA (Fixed Address) button is depressed (lit up). The alpha-numeric information used by the authors was the title of the record.

The last four characters contain the cursor I.D. (identification) to be assigned to the operator button being

Sample Tape Record

5040+21460+12020PACOIMA DAM N76W1111

FIGURE 10. ELECTRAK CONSOLE

used and is written into the display by pressing the buttons marked BI with the CI button depressed (or lit up).

Figure 11 shows the Tape Drive Unit. The operation of the TRAK 100 is as follows:

On Tape Drive

A. Turn the machine on.

B. Mount the tape and follow the sequence of threading shown on the tape drive unit.

C. Press B.O.F. (Beginning of File) button. The tape will advance until it reaches a silver marker on the tape.

On Console

A. Depress the test button.

B. Dial thumbwheel digits which must be a multiple of 36. The author used 5040 because 504 words was the limiting buffer-in arrangement of words printed by the tape examine routine on the CDC 6400 computer. Each word on the CDC machine is 10 characters.

C. Depress the FA button and type in the identifying alpha-numeric title.

D. Depress the CI button (light on) and press the CI information on the console keyboard, BI. It is recommended that all four numbers be the same, and they must be placed in the leftmost positions on the display, if the main cursor button is to be used.

E. Mount the accelerogram on the table as later described.

FIGURE 11-a. ELECTRAK TAPE DRIVE UNIT

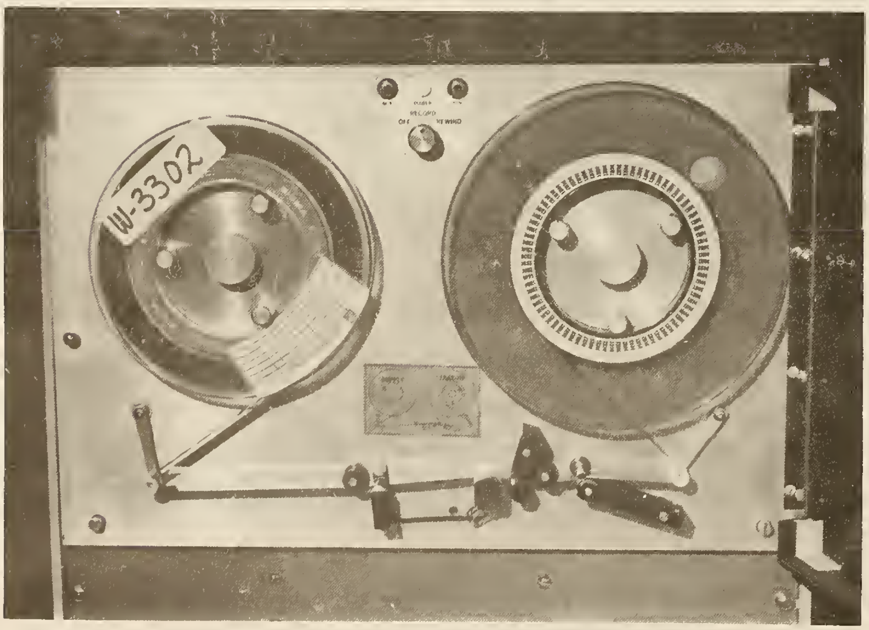


FIGURE 11-b. SEQUENCE OF THREADING TAPE, ELECTRAK TAPE DRIVE

F. Depress the test button again. Place the cursor at the position where a new origin is to be established.* Press the "0" button on the console after pressing the cursor button only once. This causes the displayed coordinates to be subtracted from all subsequent readings, thus establishing a new relative origin.

G. Place the cursor over each control point (Figure 12) and record the X and Y coordinates for future use as a check against point displacement of the record or plastic top during digitizing, or matching a correction record after readout by the computer.

H. Depress the operation button (light on). The machine is now ready for digitizing. As each point is recorded by pressing the cursor button, a short "beep" is heard by the operator as the seven track tape is recorded at the rate of 556 bits/inch.** When the end of a block of data is recorded (counter goes to zero) a longer beep is heard, signifying an inter-record gap of approximately 3/4" on the tape.

I. When one complete curve is digitized, press the E.O.F. button on the tape drive only once. The tape will record a 3/4" space for inter-record gap followed by an end of file mark, followed by another 3/4" gap before the beginning of information for a new curve.

---

* Default allows the origin to remain at the lower left corner of the table.

** First 6 bits is used to record the character information, while the seventh bit is used to check for parity errors. Even parity is recorded by the Electrak when seven-track tape is used. See page 26 for information on nine-track tape.

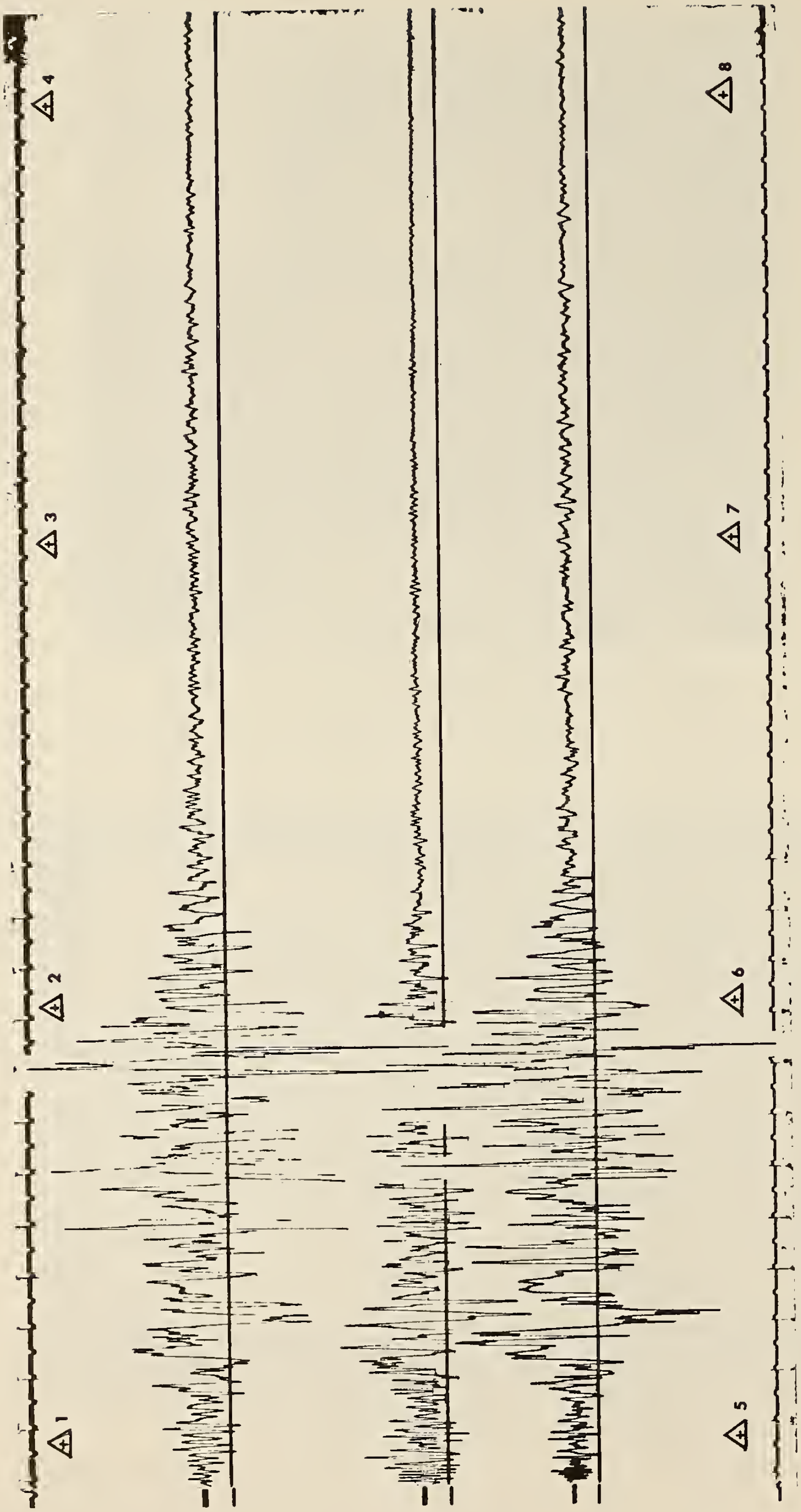


FIGURE 12. ACCELEROGRAM WITH CONTROL POINTS ADDED

J. When all the record data has been entered on the tape, press the E.O.F. button four times in succession to signify end of information. Once this has been done, no additional information may be recorded on that tape. WARNING: If the machine is turned off prior to end of information, it will not be possible to add additional data on the tape. There is no provision in the TRAK 100 for skipping tape files.

2. Mounting the Accelerogram on the Electrak Machine

Figure 12 shows a typical earthquake accelerogram recorded on film. It consists of three components of an earthquake record with a fixed trace for each record. To this accelerogram a series of eight control points represented by triangles has been added.

The accelerogram is mounted on the digitizing table so that one of the fixed traces is parallel to the lower table frame. With the test button depressed on the console the cursor is used for final adjustment of the accelerogram position on the table by reading the Y position on the console of several points along the fixed trace. As long as the test button is depressed no information will be recorded on the tape, but the console will still display coordinate data as per location of the crosshair on the cursor. When most of the fixed trace points have approximately the same Y coordinate, the accelerogram is firmly attached to the table top with masking tape along its top and bottom edges.

The coordinate location of all control points are recorded for future reference. Now one is ready to record digitized data by depressing the operate button on the console.

3. Electrak Digitizing

Figure 13 shows a sample of recorded information of coordinate data for the N76^{o}W component. This data was obtained from a seven-track tape using a special examine subroutine called TRAK 010 whose purpose was to check valid character input to the tape. Two major problems were encountered in translating the information from the seven-track tape used by the Watershed Research Station Electrak machine to the CDC computer. First, the + and - characters identifiable by the TRAK 100 were not the same for the CDC computer since the recorded 556 bits/inch data was in a modified external BCD code on the tape. Subroutine TRAK 010 forms the conversion. The second problem encountered was that TRAK 100 would drop characters occasionally resulting in a garbled field of data. The TRAK 010 examines each character in sequence on the tape and when the sequence within a 36-character stream is correct, it records the coordinates of the data point represented by the stream. Details concerning TRAK 010 are given in Appendix E.

For nine-track tapes operational on the IBM 370 computer, the Electrak records data in EBCDIC form with proper code for the + and - characters at the rate of 800 bits/inch. An alternate TRAK 010 subroutine is included in the IBM computer listing in Appendix D, however, in the later version no attempt was made to eliminate points containing garbled coordinate information, since they could provide a clue to improper digitizing. It is advisable to screen the entire list for alphabetic data. Where this occurs occasionally, the card punched data should be corrected. It is proper to duplicate adjacent data to replace erroneous information.

** PACOIMA N76W ELECTRAK **

TIME - ACCELERATION DATA READ FROM MAG TAPE

```
5040+19350+09030
5004+19370+09065
4968+19375+09085
4932+19385+09110
4896+19400+09135
4860+19410+09170
4824+19420+09115
4788+19420+09080
4752+19425+09060
4716+19430+09030
4680+19440+09070
4644+19450+09105
4608+19455+09165
4572+19455+09220
4536+19460+09285
4500+19470+09335
4464+19470+09295
4428+19480+09260
4392+19480+09195
4356+19480+09160
4320+19480+09125
4284+19490+09070
4248+19500+09110
4212+19510+09025
4176+19520+09070
4140+19525+09125
4104+19530+09175
4068+19535+09215
4032+19540+09245
3996+19545+09280
3960+19550+09215
3924+19560+09160
3888+19560+09095
3852+19560+09050
3816+19560+08995
3780+19560+08970
3744+19565+08930
3708+19570+08925
3672+19575+08975
3636+19585+09020
3600+19590+09060
3564+19590+09110
3528+19595+09145
3492+19605+09170
3456+19615+09225
3420+19620+09190
3384+19625+09170
3348+19630+09150
3312+19645+09125
3276+19650+09150
3240+19655+09180
3204+19660+09220
3168+19665+09260
3132+19670+09290
```

FIGURE 13. SAMPLE OF OUTPUT DATA LISTED BY SUBROUTINE TRAK 010

All coordinate points which are valid are then punched onto cards with a specification of 4(F10.3,F10.3) format as illustrated in Figure 14 In addition, the data is plotted to the same scale as the original curve on the accelerogram by the PLOTTR Subroutine. Figure 15 shows the Electrak plot of the digitized coordinates for the N76^{o}W component of the Pacoima Dam record.

An enlarged view of a few seconds of the Electrak record is shown in Figure 16, indicating the location of digitized points in relation to an enlarged background view of the original record. This total record contained 2628 points per 28.6 seconds of record. Figure 17 shows the digitized points recorded by Trifunac (1971) which has 2685 points per 41.7 seconds of record. Essentially in digitizing one needs to use the following guidelines:

A. Try to stay within the centerline of the curve. With a measuring magnifier the curve is enlarged to facilitate the location of the cursor directly along the middle path of the curve.

B. Care should be exercised so that points are located at peaks and valleys at the intersection of the trace centerlines.

C. More points need to be taken along the curved paths then along the straight paths of the accelerogram.

D. Points need to be located at all apparent changes in tangentlines to the center path of the curve.

Figures 18 through 21 show additional segments of the digitized record as further illustrations of the above principles.

NO OF DATA POINTS ARE 139
DATA AS IT APPEARS FROM INPUT FOLLOWS

19350.000	9030.000	19370.000	9065.000	19375.000	9085.000	19385.000	9110.000
19400.000	9135.000	19410.000	9170.000	19420.000	9115.000	19420.000	9080.000
19425.000	9060.000	19430.000	9030.000	19440.000	9070.000	19450.000	9105.000
19455.000	9165.000	19455.000	9220.000	19460.000	9285.000	19470.000	9335.000
19470.000	9295.000	19480.000	9260.000	19480.000	9195.000	19480.000	9160.000
19480.000	9125.000	19490.000	9070.000	19500.000	9110.000	19510.000	9025.000
19520.000	9070.000	19525.000	9125.000	19530.000	9175.000	19535.000	9215.000
19540.000	9245.000	19545.000	9280.000	19550.000	9215.000	19560.000	9160.000
19560.000	9095.000	19560.000	9050.000	19560.000	8995.000	19560.000	8970.000
19565.000	8930.000	19570.000	8925.000	19575.000	8975.000	19585.000	9020.000
19590.000	9060.000	19590.000	9110.000	19595.000	9145.000	19605.000	9170.000
19615.000	9225.000	19620.000	9190.000	19625.000	9170.000	19630.000	9150.000
19645.000	9125.000	19650.000	9150.000	19655.000	9180.000	19660.000	9220.000
19665.000	9260.000	19670.000	9290.000	19670.000	9305.000	19675.000	9345.000
19675.000	9310.000	19685.000	9280.000	19680.000	9230.000	19685.000	9200.000
19680.000	9175.000	19685.000	9135.000	19685.000	9110.000	19685.000	9070.000
19685.000	9050.000	19690.000	9010.000	19690.000	8980.000	19690.000	8945.000
19695.000	8915.000	19700.000	8895.000	19695.000	8870.000	19700.000	8855.000
19705.000	8920.000	19710.000	8960.000	19720.000	9030.000	19720.000	9095.000
19725.000	9150.000	19730.000	9200.000	19730.000	9245.000	19735.000	9290.000
19745.000	9330.000	19740.000	9350.000	19755.000	9320.000	19755.000	9290.000
19760.000	9250.000	19760.000	9205.000	19760.000	9170.000	19760.000	9125.000
19765.000	9080.000	19765.000	9050.000	19770.000	9020.000	19765.000	8995.000
19765.000	8975.000	19770.000	8950.000	19770.000	8940.000	19775.000	8975.000
19785.000	9010.000	19785.000	9055.000	19795.000	9095.000	19800.000	9145.000
19800.000	9180.000	19810.000	9215.000	19810.000	9240.000	19815.000	9280.000
19825.000	9310.000	19825.000	9345.000	19830.000	9315.000	19835.000	9300.000
19845.000	9270.000	19845.000	9235.000	19850.000	9200.000	19855.000	9145.000
19855.000	9080.000	19855.000	9025.000	19855.000	8970.000	19855.000	8915.000
19855.000	8865.000	19855.000	8830.000	19860.000	8805.000	19855.000	8775.000
19860.000	8730.000	19860.000	8690.000	19865.000	8675.000	19870.000	8640.000
19875.000	8680.000	19880.000	8765.000	19875.000	8815.000	19885.000	8870.000
19885.000	8920.000	19890.000	8990.000	19895.000	9070.000	19895.000	9125.000
19905.000	9200.000	19905.000	9275.000	19910.000	9340.000	19910.000	9395.000
19915.000	9450.000	19920.000	9490.000	19920.000	9525.000		

FIGURE 14. LIST OF INPUT DATA FROM SUBROUTINE PHASE1

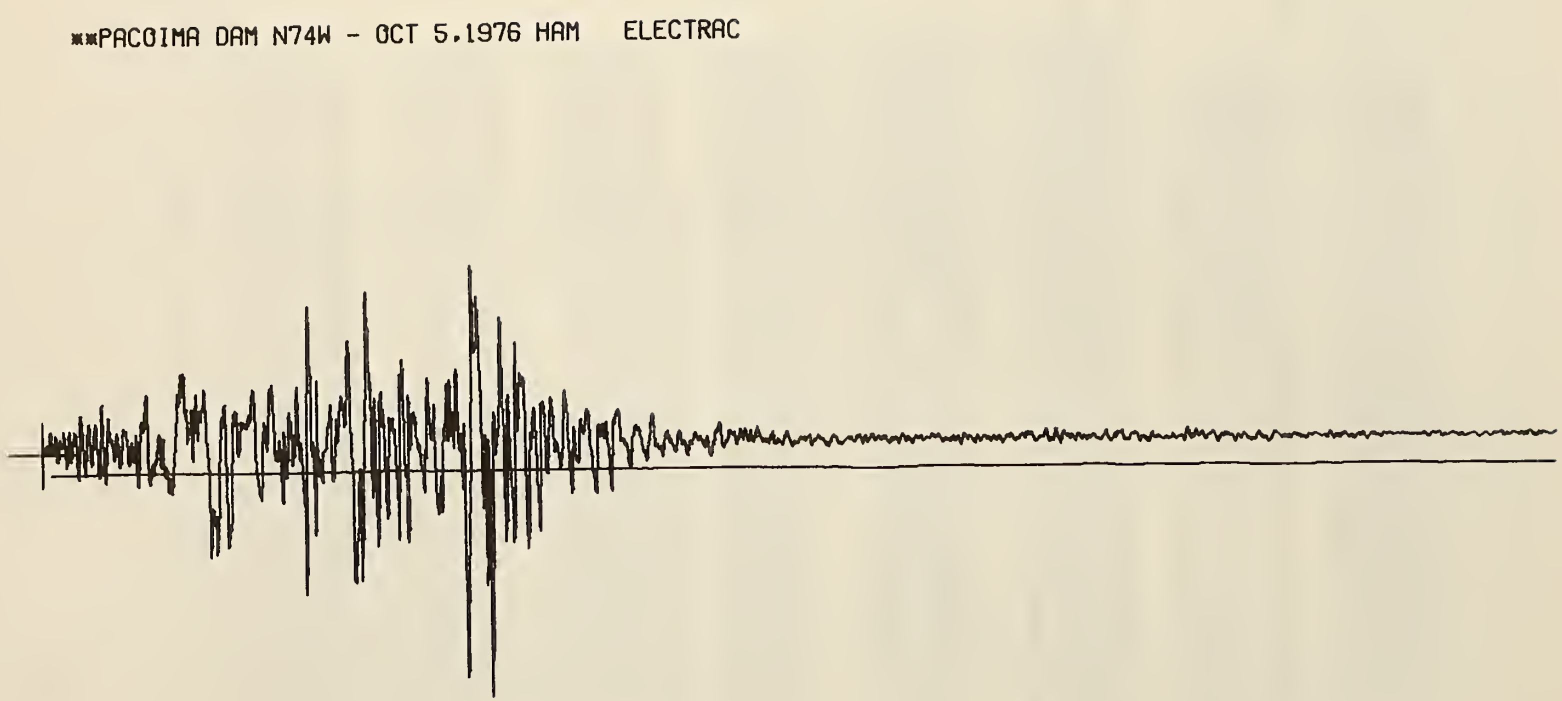


FIGURE 15. OUTPUT PLOT VIA PLOTTR SUBROUTINE, N76^{o}W COMPONENT, PACOIMA DAM RECORD

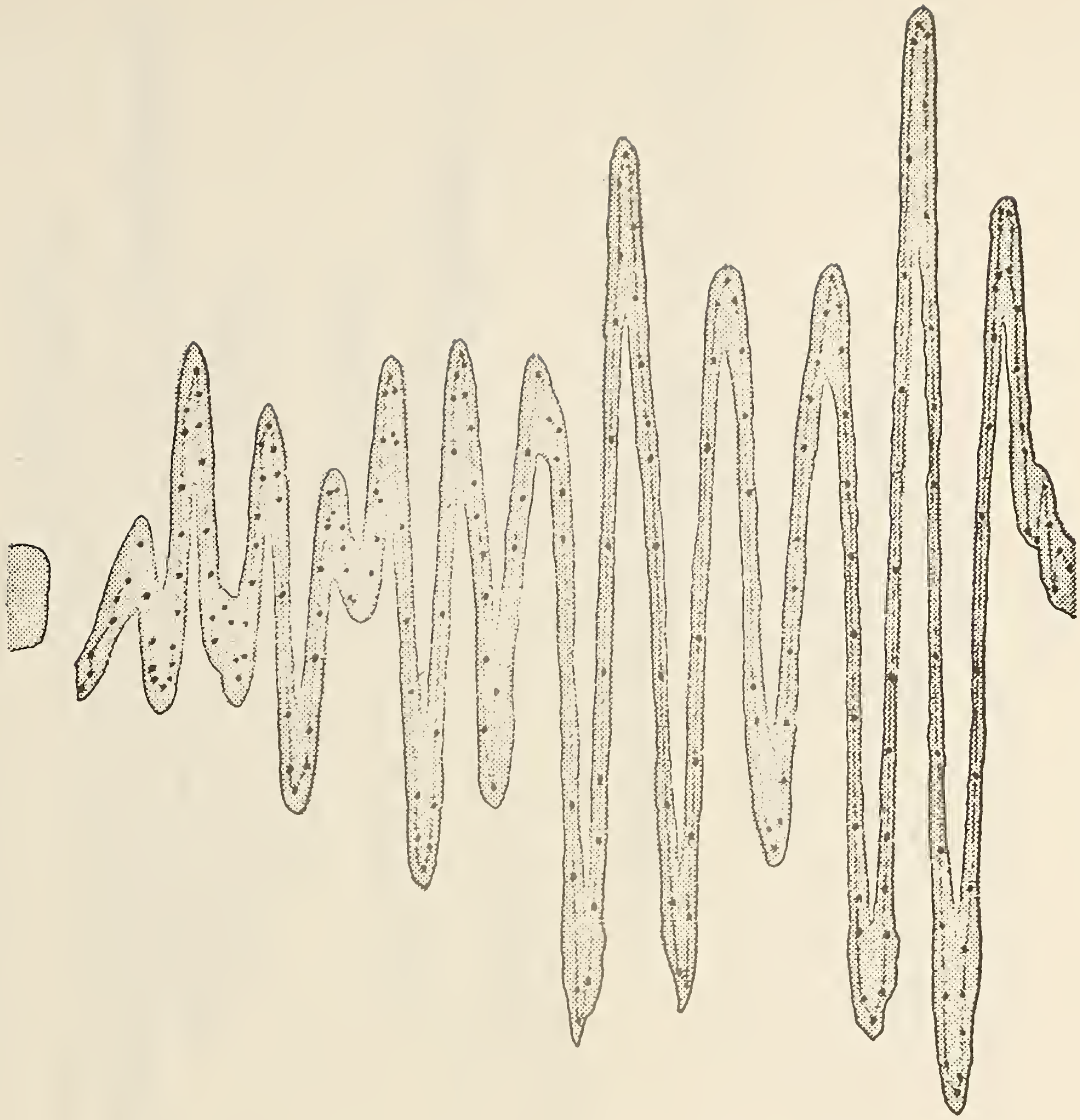


FIGURE 16. ENLARGED VIEW OF FIRST 1.3 SECONDS OF N76°W PACOIMA DAM RECORD SHOWING THE LOCATION OF DIGITIZED POINTS FROM ELECTRAK DATA

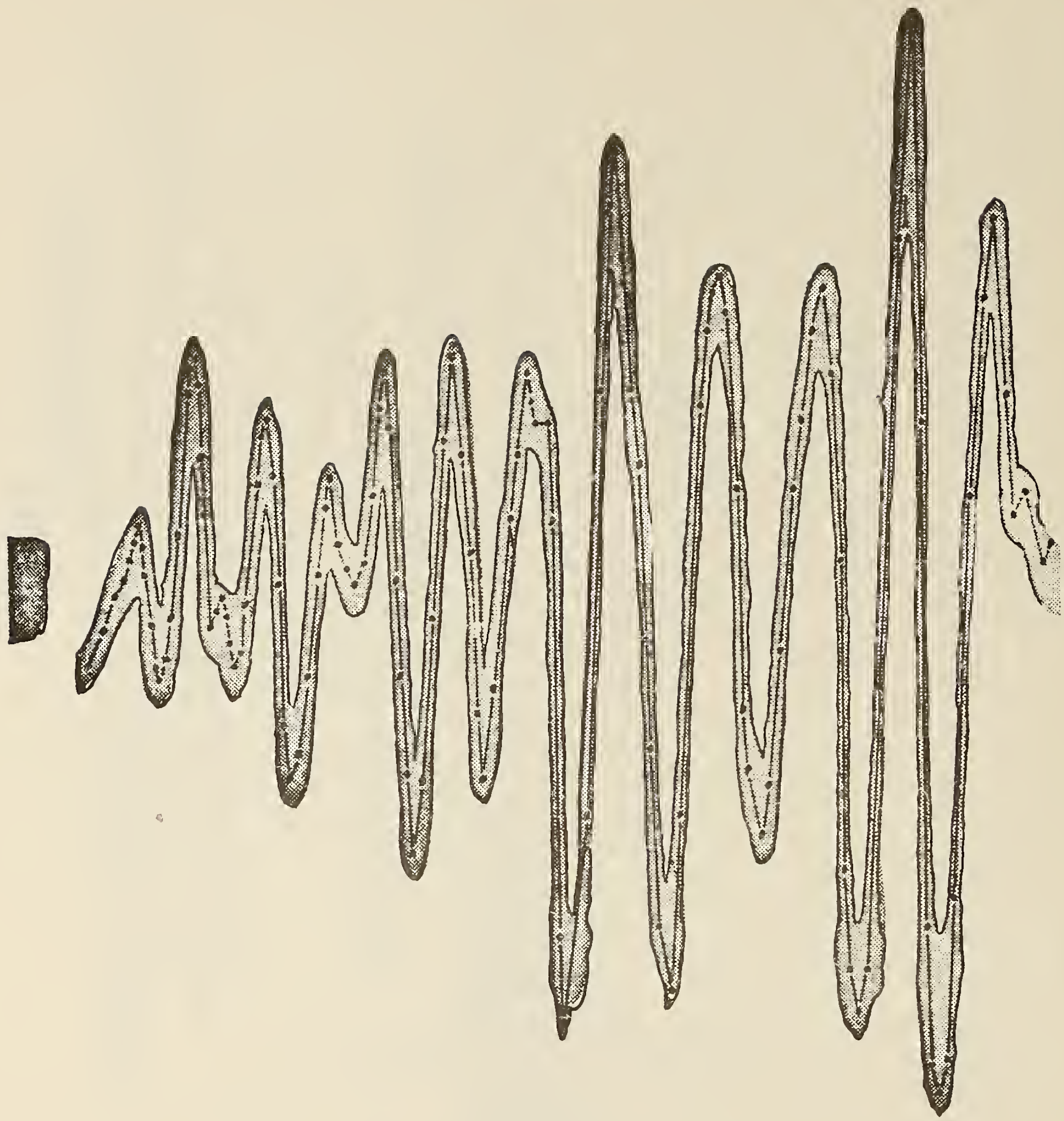


FIGURE 17. ENLARGED VIEW OF FIRST 1.3 SECONDS OF PACOIMA DAM RECORD SHOWING THE LOCATION OF DIGITIZED POINTS FROM CAL TECH DATA (8)

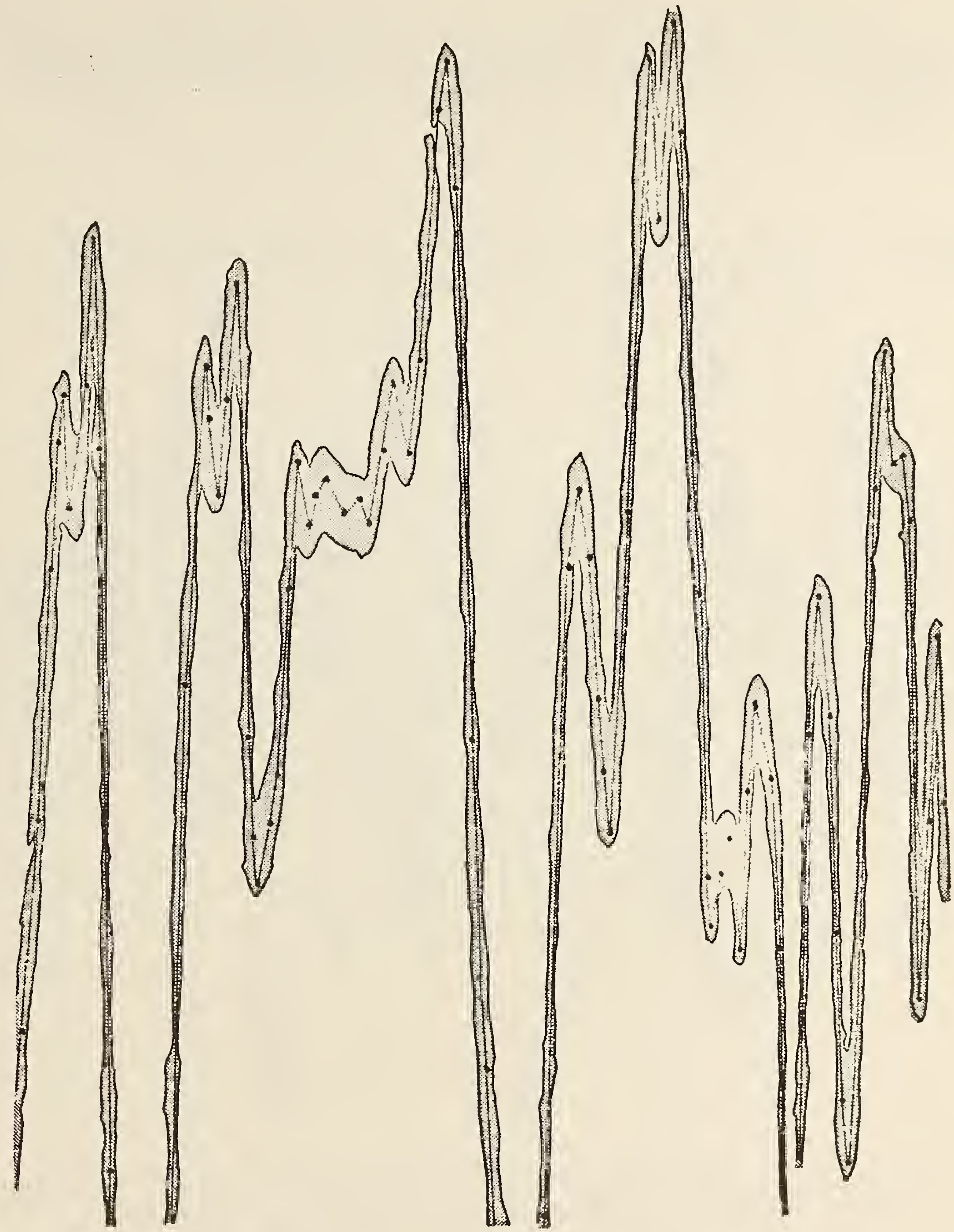


FIGURE 18. ENLARGED VIEW OF 3.3 SECONDS TO 4.8 SECONDS OF N76°W PACOIMA DAM RECORD SHOWING THE LOCATION OF DIGITIZED POINTS FROM CAL TECH DATA

FIGURE 19. ENLARGED VIEW OF 25.3 SECONDS TO 26.6 SECONDS OF N76°W PACOIMA DAM RECORD SHOWING THE LOCATION OF DIGITIZED POINTS, CAL TECH DATA

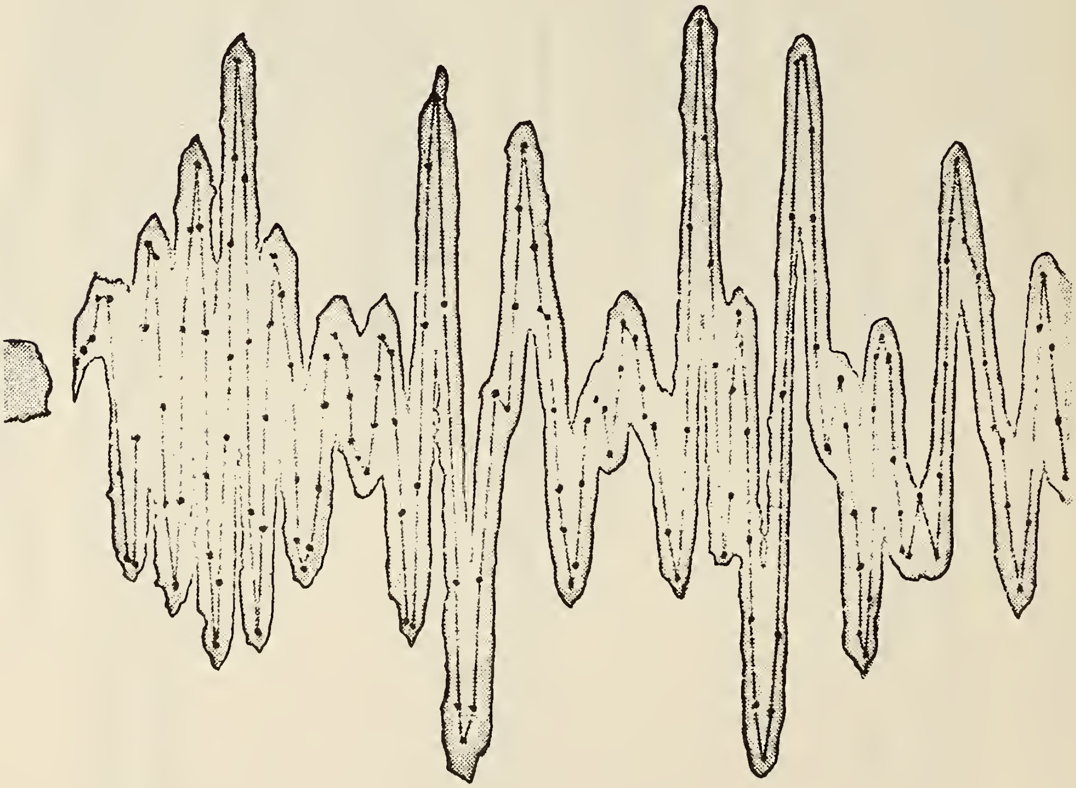


FIGURE 20. ENLARGED VIEW OF FIRST 1.3 SECONDS OF S16°E PACOIMA DAM RECORD SHOWING THE LOCATION OF DIGITIZED POINTS FROM CAL TECH DATA

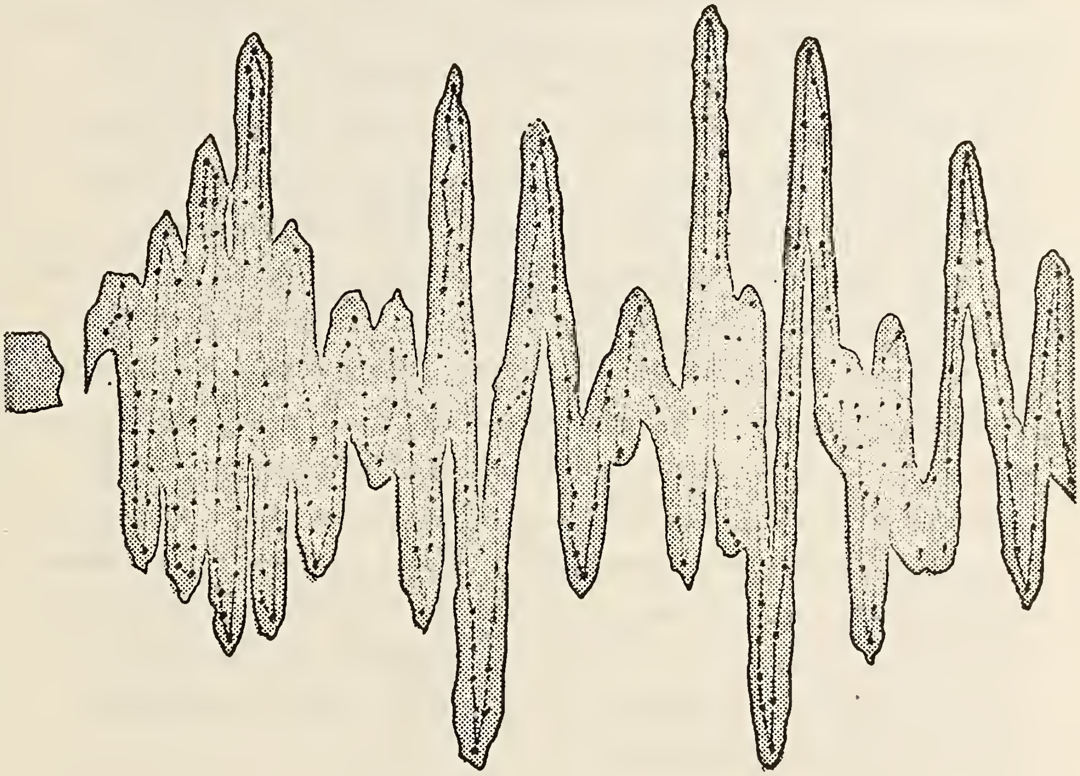
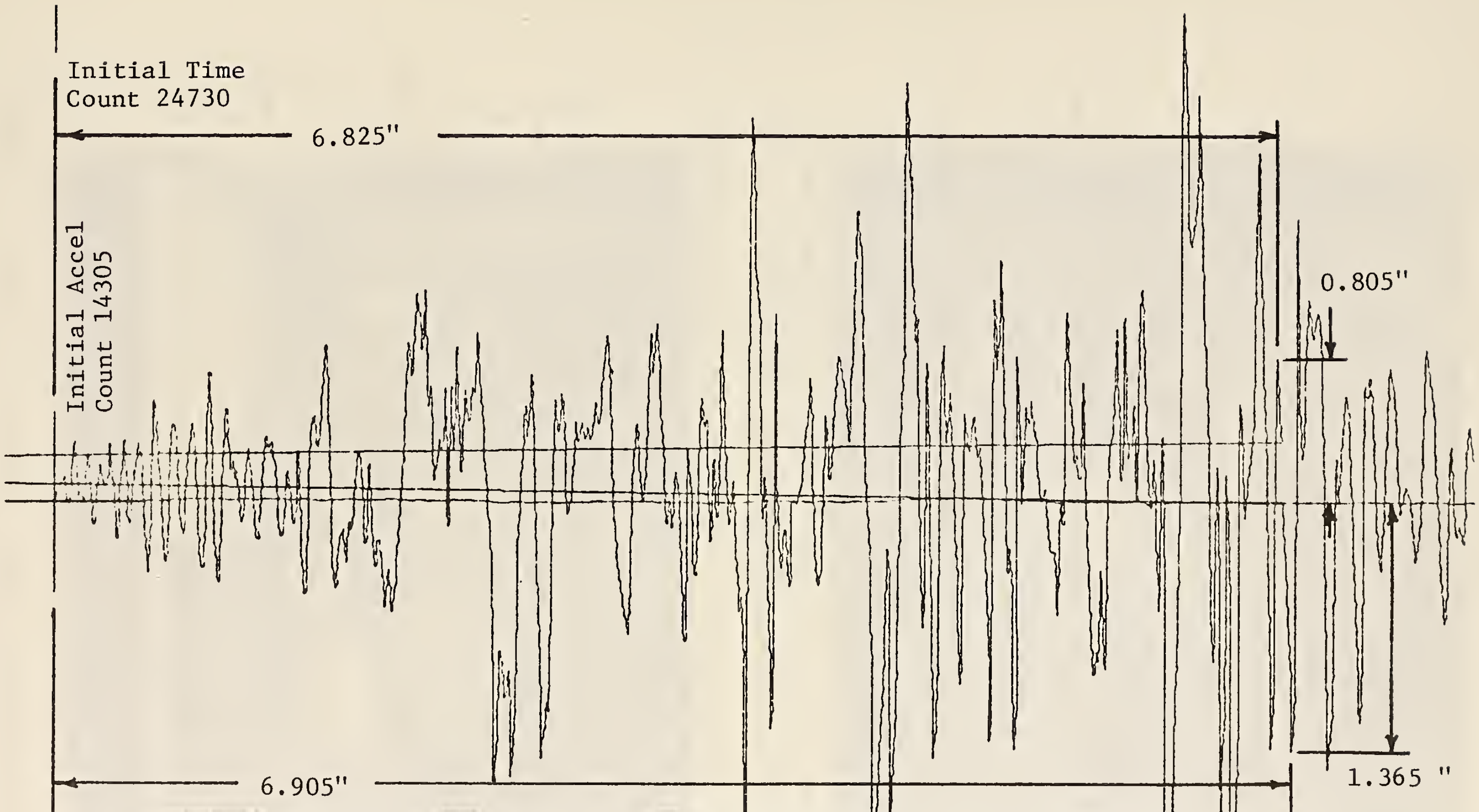


FIGURE 21. ENLARGED VIEW OF FIRST 1.3 SECONDS OF S16°E PACOIMA DAM RECORD SHOWING THE LOCATION OF DIGITIZED POINTS FROM ELECTRAK DATA

4. Digitization Corrections for the Electrak Machine

The plotted full-scale digitized accelerogram using the PLOTTR subroutine is shown in Figure 15. For checking it is overlain on the film copy of the original record and examined over a light table. It will be seen that at some points the errors will be easily discernable and are of two types: (a) points connected by a straight line different from the record indicating that intermediate digitized points were lost by the TRAK 100 machine and; (b) errors in duplicating the record by the operator. Both types of errors need to be corrected by an additional run on the digitizing machine. Prior to redigitizing, the control points should be re-examined to make sure that the location has not changed appreciably. In one run the authors found that fasteners during the servicing of the machine had been left out of the table frame which permitted the plastic table top to move relative to the wires, creating a slight translation and rotation of the record relative to a pre-established axes. For ease in locating matching end points of the correction plot with the original plot, the correction plot should be digitized starting at a readily definable peak or valley and ending at a readily definable peak or valley (Figure 22).

The PLOTTR subroutine plots the record in units of inches for both the X and Y axis. To find the X coordinate in the listing for a particular location on the record, one merely scales the differences in inches between the beginning of the record and the point in question on the plot in the X direction and then multiplies that value by 1000 and adds the initial X listed value. Remember, on the Electrak machine 1000 points equals one inch, in both the X and Y directions.



31520.000	13375.000	31520.000	13085.000	31520.000	12955.000	31525.000	13065.000
31535.000	13160.000	31540.000	13340.000	31540.000	13545.000	31540.000	13980.000
31540.000	14145.000	31545.000	14445.000	31545.000	14645.000	31550.000	14820.000
31550.000	14985.000	31555.000	$\boxed{15110.000}$	31560.000	15000.000	31565.000	14935.000
31570.000	14790.000	31580.000	14650.000	$\boxed{18580.000}$	14480.000	31585.000	14310.000
31590.000	14190.000	31595.000	14080.000	31605.000	13975.000	31610.000	13815.000
31515.000	13685.000	31625.000	13490.000	31635.000	13365.000	31640.000	13200.000
31635.000	13050.000	31635.000	$\boxed{12940.000}$	31645.000	13040.000	31655.000	13265.000
31660.000	13505.000	31655.000	13335.000	31655.000	14070.000	31665.000	14245.000
31665.000	14515.000	31665.000	15030.000	31665.000	15375.000	31660.000	15715.000
31660.000	15880.000	31670.000	15690.000	31675.000	15455.000	31680.000	15170.000

FIGURE 22. CORRECTIONS TO DIGITIZED DATA, ELECTRAK MACHINE

5. Benson-Lehner Reader and Decimal Converter

A Benson-Lehner Reader and Decimal Converter at the Watershed Research Station was also used as a digitizing machine for earthquake records. Essentially, this machine is classified as an "analog-to-digital" reduction device which is used to digitize coordinate values of an accelerogram record. The machine consists of three major components. Figure 23 shows the chart reader (Model E), Figure 24 shows the IBM-26 card punch machine, and Figure 25 shows the decimal converter (Model F). All components are electronically interconnected to facilitate the output of the digitized record in punched card form. The reader electronically "reads" X and Y coordinates along the trace. The decimal converter then interprets the data so that it may be output as a number of counts above and to the right of a specific reference point. This information is then conveyed to the IBM card punch where the X and Y coordinates are automatically punched on cards.

6. Theory of Operation for the Benson-Lehner Machine

The reader, as shown in Figure 26, basically consists of an inclined light table on which the accelerogram is mounted, and two large knobs, (A and B) which control the location of the X and Y coordinates. Essentially these knobs connected to potentiometers, mechanically control the movements of a vertical reference line "D" (VRL) and an inclined calibration line "E" (CL). Knob A controls the horizontal position of the VRL from the established zero reference origin in the X direction. Knob B controls the horizontal distance IJ within the range of the plastic overlay. Once the angle of the calibration line E is set, all vertical positions KJ are proportional to the

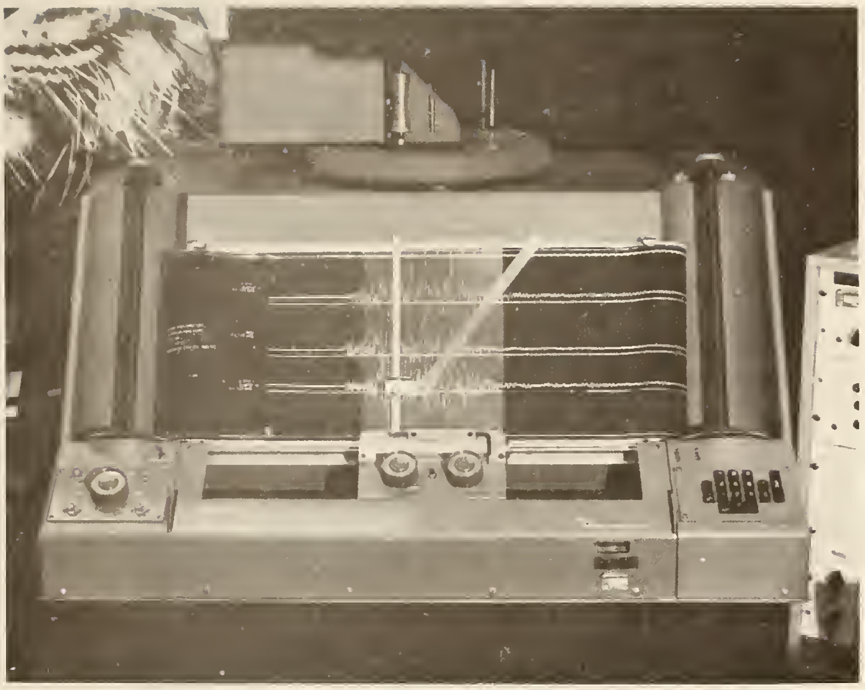


FIGURE 23. BENSON-LEHNER CHART READER

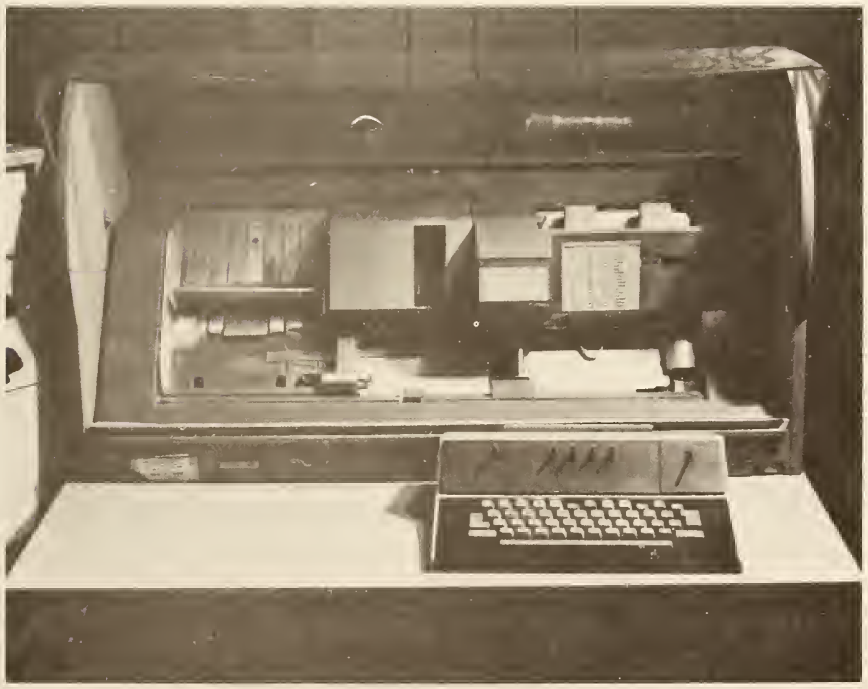


FIGURE 24. IBM-26 CARD PUNCH MACHINE FOR BENSON-LEHNER READER

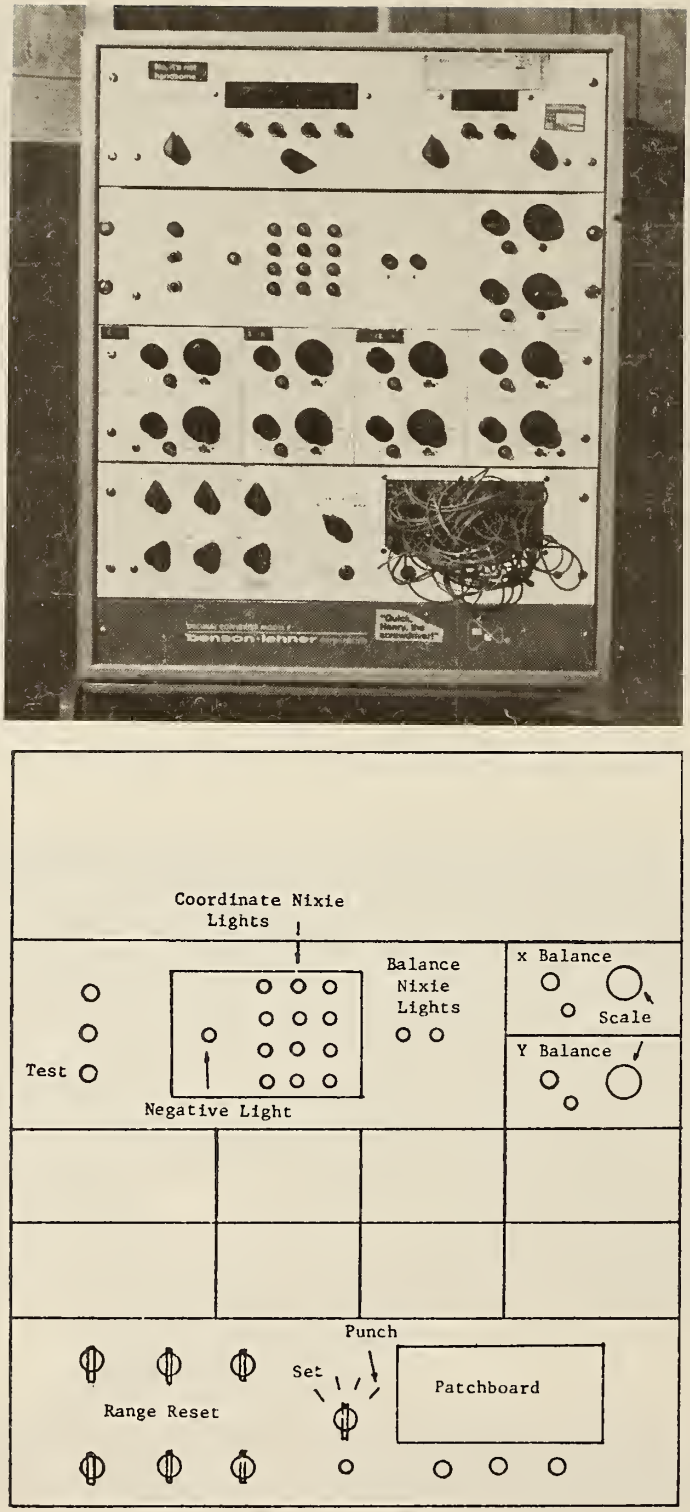
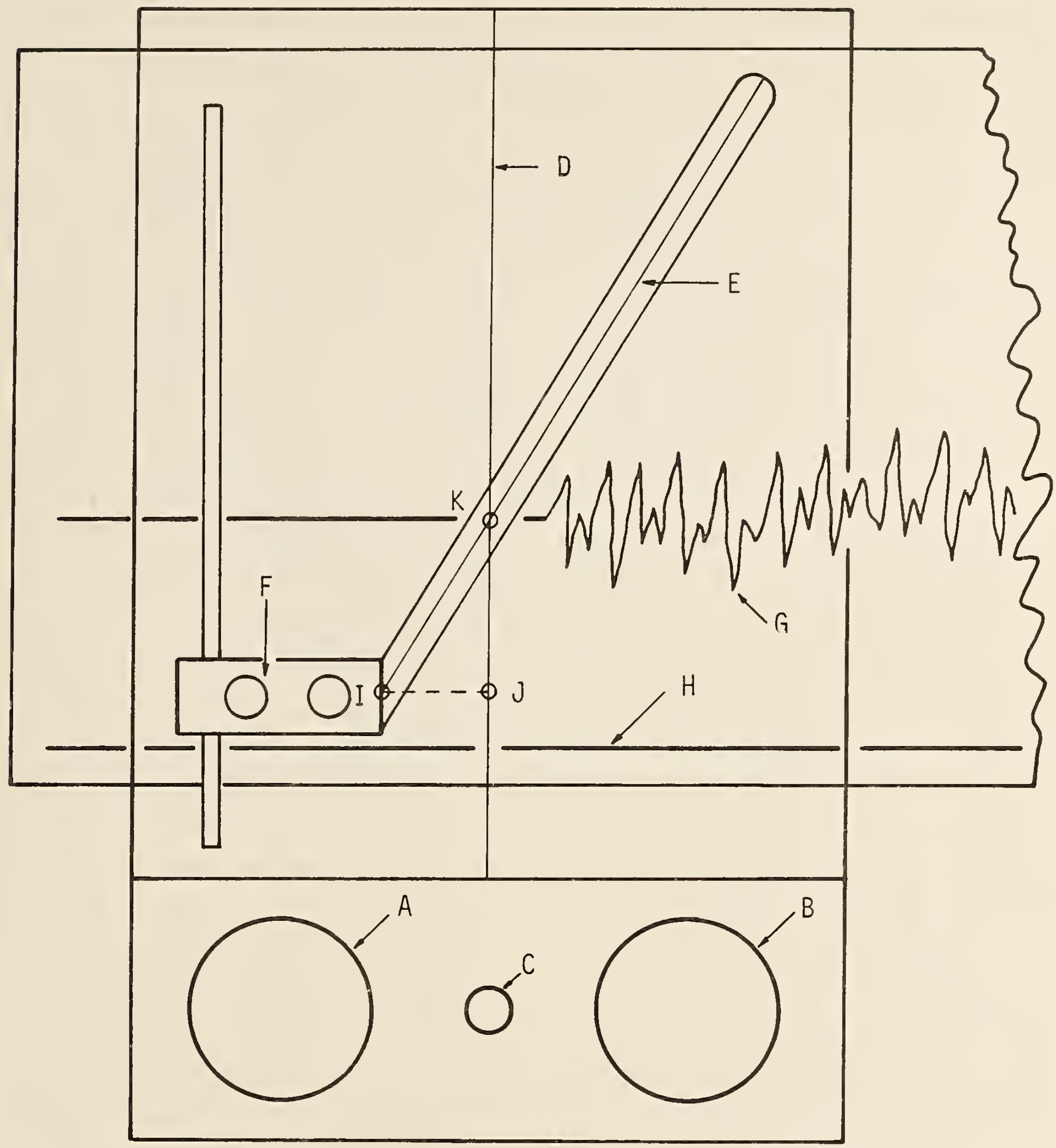


FIGURE 25. BENSON-LEHNER DECIMAL CONVERTER



A. Left Control Knob
B. Right Control Knob
C. Scan (Read-Out Button)
D. Vertical Reference Line (VRL)
E. Calibration Line
F. Overlay Clamp
G. Accelerogram Trace
H. Fixed Trace

FIGURE 26. BENSON-LEHNER CHART READER SET-UP POSITION

corresponding horizontal distance IJ. Thus two coordinates, X and Y, are determined from horizontal positions of the VRL and CL. When these two lines are positioned to intersect directly over a point "K" to be digitized, a read-out button "C" is depressed and an electrical impulse is sent to the decimal converter. These impulses are actually a measure of both potentiometer resistances which are related to the relative position from an established origin to the VRL and CL. These input resistances are measured by a ratio bridge which is designed to sense the position of the input potentiometers.

The decimal converter receives these input resistances and converts them to "counts", i.e., a proportional part of the total resistance. The circuit of the converter is designed to scan a series of coded resistors, compare them with the incoming resistance, balance and hold or drop them depending upon whether an overbalance is sensed. When the input resistance is balanced with the internal resistance, a null is reached. If there is an incorrect balance or if there are input changes during this balancing procedure, an error is detected and further processing is halted.

When the input resistance is balanced with the coded decimal resistances, an impulse is conveyed to the IBM card punch. The format of the readout sequence is controlled by a patchboard which is adopted for computer input.

The patchboard contains hubs (Figure 27) into which wires are plugged. There are a total of sixty (60) positions available for readout classified as Mode No. 1 or Mode No. 2 output. These positions are scanned when the record scan button is depressed, thereby allowing the card punch format to be completely controlled by the patchboard setup.

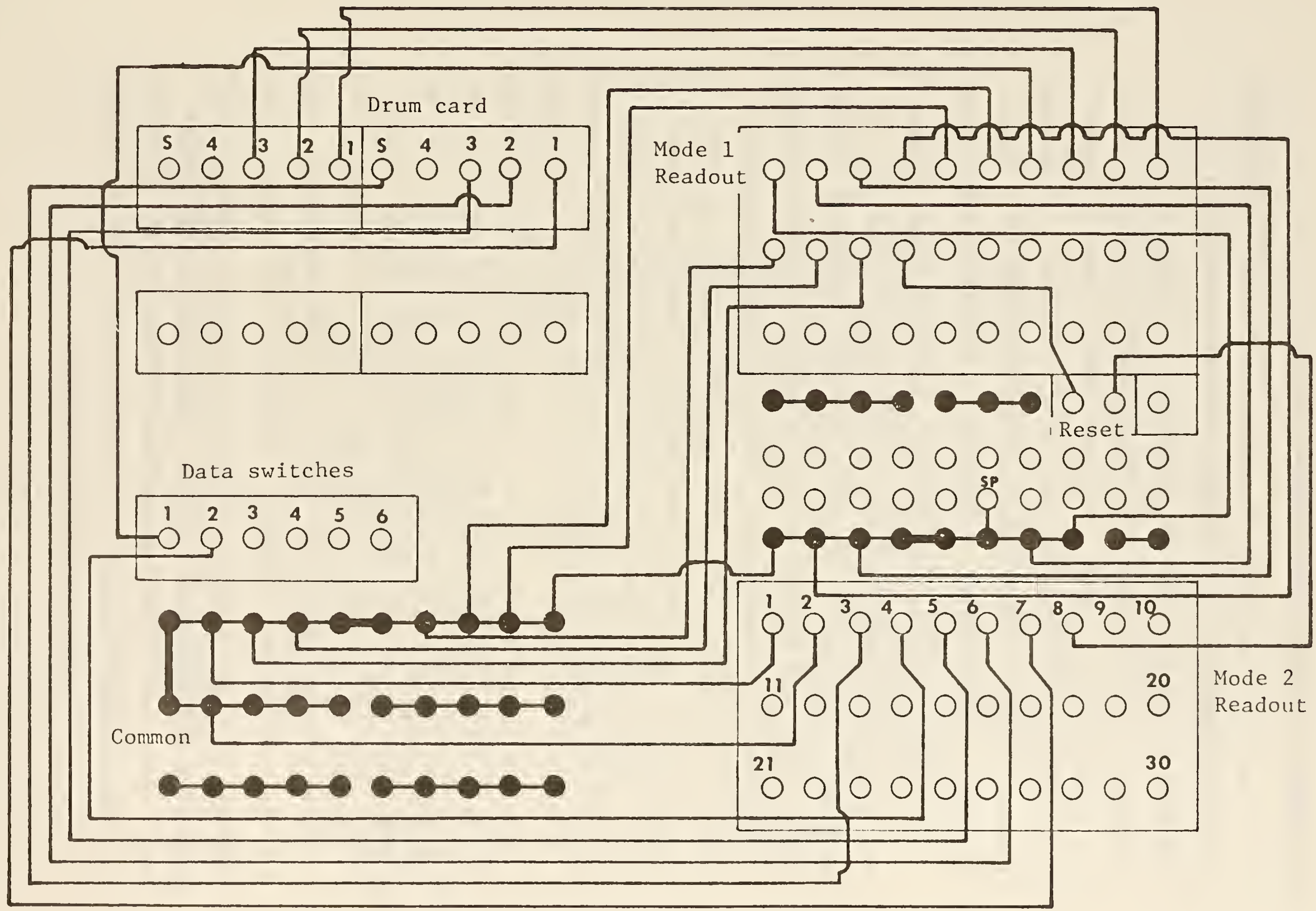


FIGURE 27. BENSON-LEHNER DECIMAL CONVERTER PATCHBOARD

7. Patchboard Wiring of Benson-Lehner Machine

As shown in Figure 27, there are several boxed areas on the patchboard. Certain areas are used for specialized formatting; however, for the purposes of this report it is necessary to use only the following areas: 1) X ANALOG area, 2) Y ANALOG area, 3) MODE 1 READOUT area, 4) MODE 2 READOUT area, 5) DATA SWITCHES area, and 6) the "common" area denoted by open circles. Common hubs merely act to duplicate input characters (e.g., numeric digits, +, -, ", blank spaces, etc.).

When a balance resistance is met, an output impulse is sent to scan the Mode 1 readout (when the mode switch is in the Mode 1 direction). Beginning from the first hub of Mode 1 readout area, the machine scans the hubs and outputs whatever is "patched" into these hubs. Each mode hub represents one column on a computer card. For example, if hub 1 of Mode 1 readout area is patched to Data Switch 3, whatever number data switch 3 is on, that number will be sent to the card punch and will be punched in column 1. If the second hub is patched to a special character, e.g., "+", a "+" sign will be punched out in the second column of the computer card. The patchboard on the Model F converter has the capacity to output up to 60 characters per scan (data point). Since the authors used the format 2I10 to define the X and Y coordinate values at each point, only 20 characters were used. That is, the data on the computer card appeared like this: ssssssxxxxssssss±yyy. The s's represent spaces (blanks), followed by four digits representing the X coordinate, followed by six more spaces, a sign for the Y coordinate, followed by three digits representing the Y coordinate. The authors found it convenient to utilize the MODE 1 READOUT for the X coordinate and the MODE 2 READOUT for the Y coordinate as further explained.

To wire the patchboard for the 2110 format, the following steps are used:

1. *Wire the space hub (SP) to the common hub as shown in Figure 27, so that the twelve vacant hubs are made available for the twelve spaces of the output.
2. Wire the first six hubs of MODE 1 to six of these common spaces mentioned in Step 1.
3. Wire hub 7 of MODE 1 to data switch 1. This will output the digit selected by data switch 1 in the seventh column of the punched card.
4. Wire hubs 8, 9, and 10 of MODE 1 to hubs 3, 2, and 1 respectively of the X ANALOG area. This will output the last 3 digits of the X coordinate value on the 8th, 9th, and 10th column of the punched card.
5. Wire hubs 11, 12, and 13 of MODE 1 to three more common space hubs. This puts three spaces on the punched card in columns 11, 12, and 13.
6. Wire hub 14 to the left reset hub located just below the MODE 1 area. This switches control to the MODE 2 READOUT area and the readout SCANNER will continue to scan from the first hub of MODE 2, without changing the position of the mode switch on the face of the converter.

---

* The authors believe that these instructions hold for all Benson-Lehner machines. However, the reader is advised to check his instruction manual for variations.

7. Wire the first two hubs of MODE 2 to two remaining common space hubs. This places two more spaces (blanks) in columns 14 and 15.
8. Wire hub 3 of MODE 2 to the "sign" hub of the Y ANALOG area. This places the proper sign in the 16th column of the punched card.
9. Wire hub 4 to data switch 2.
10. Wire hubs 5, 6, and 7 of MODE 2 to hubs 3, 2, and 1 respectively of the Y ANALOG area. This will output the last three digits of the Y coordinate value on the 18th, 19th, and 20th columns of the punched card.
11. Wire hub 8 of MODE 2 to the right reset hub. This switches control from MODE 2 back to MODE 1. Further scanning stops at this point until the scan (readout) button is pressed again.

When the scan button is pressed, and the mode switch is in the MODE 1 position, the readout scanner will begin to scan the MODE 1 READOUT hubs and will output whatever digit is patched into that hub. This cycle is repeated each time the scan button is pressed.

Although other wiring systems can be used to obtain the same type of output format, the authors found this system very satisfactory and did not pursue any others.

The data switch is used to output the thousands digit of the X coordinate. Since the Model F converter has a positive range of from 0 to +999, the data switch* is used to increase this range up to 9999. To accomplish this, the data switch is initially set to zero. When the X coordinate (VRL) reaches +999, the data switch is set to "1" and the X channel of the converter is nulled. This is done by pressing the clear button, and then turning the X channel origin dial until both "null"

* Range reset switch, Figure 25.

Nixie tube lights are out. Pressing the scan button will now output 1000 for the X coordinate value. The "1" (first digit) is from the data switch and the three zero's represent a new origin of the X axis. This process is repeated each time the X coordinate values reach 999; the data switch advancing to 2, 3, 4, etc., as required by the length of the accelerogram.

This process is not necessary for the Y coordinate since with the minus sign, the range is from -999 to +999. This allows 1998 counts or about 8 centimeters from the lowest to the highest point on the accelerogram. This is usually adequate for most earthquake records. A data switch can be used, however, as in the X coordinate, if this range is exceeded.

8. Mounting the Accelerogram Record on the Benson-Lehner Machine*

Mounting the accelerogram onto the inclined light table of the record reader is the first step in obtaining digitized data from an accelerogram record. Using Figure 26 as a guide, the procedure is as follows:

1. Turn on the light behind the light table.
2. Position the VRL as far to the left as possible with the left control knob (A).
3. Place the accelerogram record between the light table and the VRL. The leftmost starting point on the accelerogram (time = 0) should be approximately 1/4" to the right of the VRL and approximately centered between the top and bottom of the

---

* It should be noted that the terms "record" and "accelerogram are used synonymously in this section of the report.

light table. Temporarily hold the accelerogram in this location by placing the magnetic bars over the left and right edges.

4. Place the calibration line wand (E) in the overlay clamp (F) initially at a 45° angle.

5. Turn the right control knob (B) clockwise as far as possible. The intersection of the VRL (D) and the calibration line (E) should now be at least 1/4" below the lowest point on the accelerogram record (G) or fixed trace (H).

6. Turn the right control knob (B) counter-clockwise as far as possible. The intersection of the VRL (D) and the CL (E) should now be at least 1/4" above the highest point on the accelerogram (G). If not, adjust the angle of the CL wand (E) so that this requirement is met. Return to Step 5 and recheck the lowest point again with the new wand angle. (The overlay clamp can also be moved up or down to assist in achieving Steps 5 and 6.)

7. Turn the right control knob (B) so that the intersection location of the VRL (D) and the CL (E) is over the leftmost portion of the fixed trace (H).

8. Turn the left control knob (A) clockwise slowly and check to see that the intersection location follows the fixed trace. If the fixed trace veers away from the intersection location, tilt the accelerogram so that the intersection location follows exactly along the fixed trace. Return to Step 5 and check again to see that all the following steps (Steps 6 through 8) are carried out.

9. Using masking tape, tape the accelerogram directly to the light table so that its position cannot be altered. The accelerogram is now ready for digitization and should not be removed or repositioned until a satisfactory record* of digitized points is obtained on punched cards for the component of the accelerogram in question including the fixed trace.

9. Digitization Operation on the Benson-Lehner Machine

To operate the Benson-Lehner machine and effectively digitize coordinate data from an accelerogram, the following procedure is used:

1. Turn on the power to all three components, i.e., the light behind the reader, the decimal converter, and the card punch machine.
2. Turn the output control switch of the decimal converter to "SETUP" position to check for proper accelerogram mounting and alignment.
3. Turn both scaling knobs of the X and Y channels to the maximum capability of the machine (250 counts per cm on the Model F).
4. Press the X channel Nixie light tube (light on) to scan the X coordinate values. Turn the left control knob (A) of the reader so that the VRL (D) is as far left as possible. The location of the first point in the accelerogram should be about 1/4" right of the VRL.
5. Press the clear button and adjust the origin dial of the X channel so that the null Nixie lights are

* Including corrections after plotting, using PHASE1.

off. This sets the X origin at the intersection location, thus making all subsequent values of X positive.

6. Press the Y channel Nixie light tube (light on) to scan the Y coordinate values. Turn the right control knob (B) so that the intersection location is at the lowest point on the accelerogram trace (G).

7. Press clear button and "Force in" the digits -999 on the light bank by pressing these lights. (Lights will go on.) Now adust the origin dial of the Y channel so that the null lights are both out. This places the zero value of Y 999 counts <u>above</u> the intersection location thus leaving another 999 counts <u>above</u> the zero value of Y. Press scan to see if -999 reproduces itself.

8. Obtain the scaling factors which convert decimal converter "counts" to units of gravity and seconds. To do this, measure the vertical distance (in centimeters) between the lowest and highest point on the accelerogram, (ΔD). When the VRL and the CL intersect over each of these points, press the readout button (C) and record their respective values that appear on the Nixie light tubes. The difference between these two distances is denoted ΔV. Thus the vertical scale is $\frac{\Delta V}{\Delta D}$ counts per cm. The horizontal scale is found in the same manner using the initial time (X coordinate) value and the +999 time value as differencing points. (This information is used later to convert counts to

units of gravity and seconds and also in the PLOTTR subroutine to reproduce a full-scale plot of the accelerogram.)

9. Press the X channel Nixie light switch and check to see that the mode switch is in the MODE 1 position.
10. Turn the output control switch to "PUNCH".
11. Depressing the readout button (C) will now read the coordinate data of the intersection location of the VRL (D) and the CL (E).

This information is converted to a digital record of the X and Y coordinates and punched on computer cards.

Digitizing from here on is a simple, but tedious, operation. However, the care taken with this procedure is directly related to the quality of the final result. To achieve this objective, the operator should follow these few quidelines:

1. Move the left control knob (X coordinate) clockwise as little as possible between successive points. This allows for the maximum number of data points to be digitized.
2. Keep the intersection location as near as possible to the center of the trace, making certain to digitize locations at all peaks and valleys and all changes in straight-line patterns.
3. The eyes of the operator should always be normal to the light table and over each digitized point. One operator should digitize a complete record.

Deviation from this guideline will yield localized errors which are difficult to find and correct.*

4. Allow the converter to complete its conversion operation before moving on to another point.

These procedures and guidelines should yield data that is as reliable as possible to achieve with the Benson-Lehner digitizing machine.

10. Digitization Corrections - Benson-Lehner

After the accelerogram and fixed traces are digitized, the punched cards contain the data for use in the PHASE1 program. One of the features of this program, through the use of the PLOTTR subroutine, is to plot a full-scale reproduction of the digitized data. These plots are used to locate and correct any accidental errors in digitizing. This can be accomplished by the following procedure.

Place the full-scale plot of the digitized accelerogram over the original accelerogram record which should still be mounted on the record reader. If these traces deviate appreciably, corrections must be made. In general, this procedure serves to relate to the user the relative quality of digitization. Tolerable deviations are left to the judgement of the user. Gross errors such as peaks and valleys that are omitted appear as a straight-line connection between adjacent points.

If a correction is desired, only the portion of the accelerogram which is in error need be redigitized. This is accomplished by following the steps in the Digitization Operation section with a few exceptions. In steps 3, 4, and

* The use of a measuring magnifier as for the Electrak machine will help to define the point in the center of the curve path.

5 the null dial must be adjusted so that the initial X value and the maximum and minimum values of Y agree with those in the initial mounting procedure. In addition to the portion of the accelerogram that is to be corrected, the minimum and maximum values of the X and Y coordinates must be digitized (an additional four points). This is necessary to obtain a plot of the corrected data to the proper scale.

When the redigitized data is plotted, the corrected portion is placed over the original accelerogram and the first plot. If the two plots now appear to be a satisfactory duplicate of the original accelerogram, the incorrect data points (punched cards) are simply replaced by the new correct data. Recall that the PLOTTR subroutine plots the record in units of inches for both the X and Y axis. To find the X coordinate in the listing for a particular record, one merely scales the differences in inches between the beginning of the record and the point in question - the plot in the X direction - and then multiplies that value by the X scaling factor in counts/inch and adds the initial X listed value. A plot of the new data deck will now yield a corrected accelerogram. This full-scale plot assures the user that the data that will be used in the PHASE2 program (see Section V) is a correct representation of the accelerogram record.

11. Comparison of Electrak and Benson-Lehner Digitizers

In the previous portion of this section, two methods of digitizing an accelerogram have been presented. There are distinct advantages and disadvantages to each method. Overall, digitization with the Electrak seems to offer the best results as far as obtaining an accurate representation of the accelerogram. However, certain factors such as cost and availability of equipment can necessitate the use of the Benson-Lehner machine.

The Electrak machine has a distinct advantage over the Benson-Lehner in resolution, i.e. number of counts per linear measurement of accelerogram. The Electrak can record 1000 counts per inch as compared to the 800 counts per inch (maximum) returned by the Cal. Tech. Benson-Lehner machine. This advantage leads to a more accurate digital representation of the accelerogram.

Another primary advantage the Electrak possesses over the Benson-Lehner is in the speed of recording. It was established that the time necessary to set up and record 28 inches of accelerogram trace (including a fixed trace) on the Electrak is about four hours, while the Benson-Lehner required seven hours for 22 inches of record (including fixed trace). The primary reason for this is that each coordinate of the trace must be read, converted and punched on cards (on the Benson-Lehner) before the user can proceed to the next point. This operation takes about four seconds for each digitized point. On the Electrak machine, this is done almost instantly. (In fact, the Electrak can record continuous points at the rate of about five points per second; however, it is not physically possible to move the cursor over the accelerogram trace that fast and retain the degree of accuracy necessary for integration.)

A third advantage the Electrak maintains is in the recordable size of the accelerogram. The Electrak recording table has the capacity of recording accelerograms up to 48 inches long while the Benson-Lehner is limited to accelerogram lengths of about 22 inches.

The Electrak machine has an initial cost of around $20,000 (depending on the size of the table and peripheral equipment) while the Benson-Lehner original cost was about

half this amount. (If, of course, both machines are available to the user, original cost would not offer any advantage.)

The Benson-Lehner, although a more "primitive" analog-to-digital reduction device, offers the advantage of assurance that each coordinate point is recorded, regardless of the accuracy. Although the Electrak offers many distinct advantages over the Benson-Lehner, the user can never be absolutely certain that the recorded information is retrievable in a usable form. It was found that 95% of all digitized data was retrievable from the Electrak machine. However, in a very few cases, (due to operator error or machine error) complete records were lost and irretrievable. In some cases (due to machine error), characters containing digitized coordinates were lost, thus eliminating the coordinate point. These record loses are not realized until after the digitizing process is complete and a full-scale plot is prepared.

With the Benson-Lehner digitizing machine, however, the operator knows immediately if a certain coordinate point is not recorded. Each point is immediately punched on a computer card, the operator hears the punching process and has physical evidence (the cards) that the recording process is taking place. With the Electrak machine, the data is sent to a magnetic tape (a relatively silent process) and the quality of the information cannot be periodically inspected.

The frequency of other types of human errors appear to be equal for both digitizing methods. It should be emphasized that the reliability of the Electrak method is usually very high and is discussed above only to inform the user of the remote possibilities of record loses. Overall, the Electrak machine possesses the greatest advantage of the

digitizing process and is recommended for the digitization of accelerograms.

12. Baseline Corrected Data - Electrak versus Benson-Lehner

The resulting digitized baseline corrected accelerograms* are shown in Figures 28 and 29. Figure 28 is a plot of the baseline corrected accelerogram digitized on the Electrak machine and Figure 29 is a plot of the same baseline corrected accelerogram digitized on the Benson-Lehner machine. (Note, since the data has not been scaled (to be done in Phase2), the values representing time and acceleration are relative values only.)

At first glance, both plots appear to be quite similar. However, a closer inspection will reveal slight differences. The major difference lies in the overall number of points that were digitized. The Electrak machine recorded a greater number of points than the Benson-Lehner machine. This higher frequency of data points produces a more staggered line between adjacent peaks and valleys.

The two digitized and plotted records do, however, represent the original accelerogram in digital form. The data used to plot these graphs is now ready for further processing.

13. Perkin-Elmer Microdensitometer

Figure 30 shows the model 1010A Microdensitometer unit of the Perkin-Elmer Photometric Data Systems' Microdensitometer System,** at the Optical Sciences Laboratories of the University of Arizona. Because the system was not fully

---

* Baseline corrected means that the Y position of the fixed trace has been subtraced from the Y position of the acceleration trace using an interpolation scheme in the PHASE1 subroutine.

** Perkin-Elmer Corporation, Boller & Chivens Division, 916 Meridian Avenue, South Pasadena, California 91030.

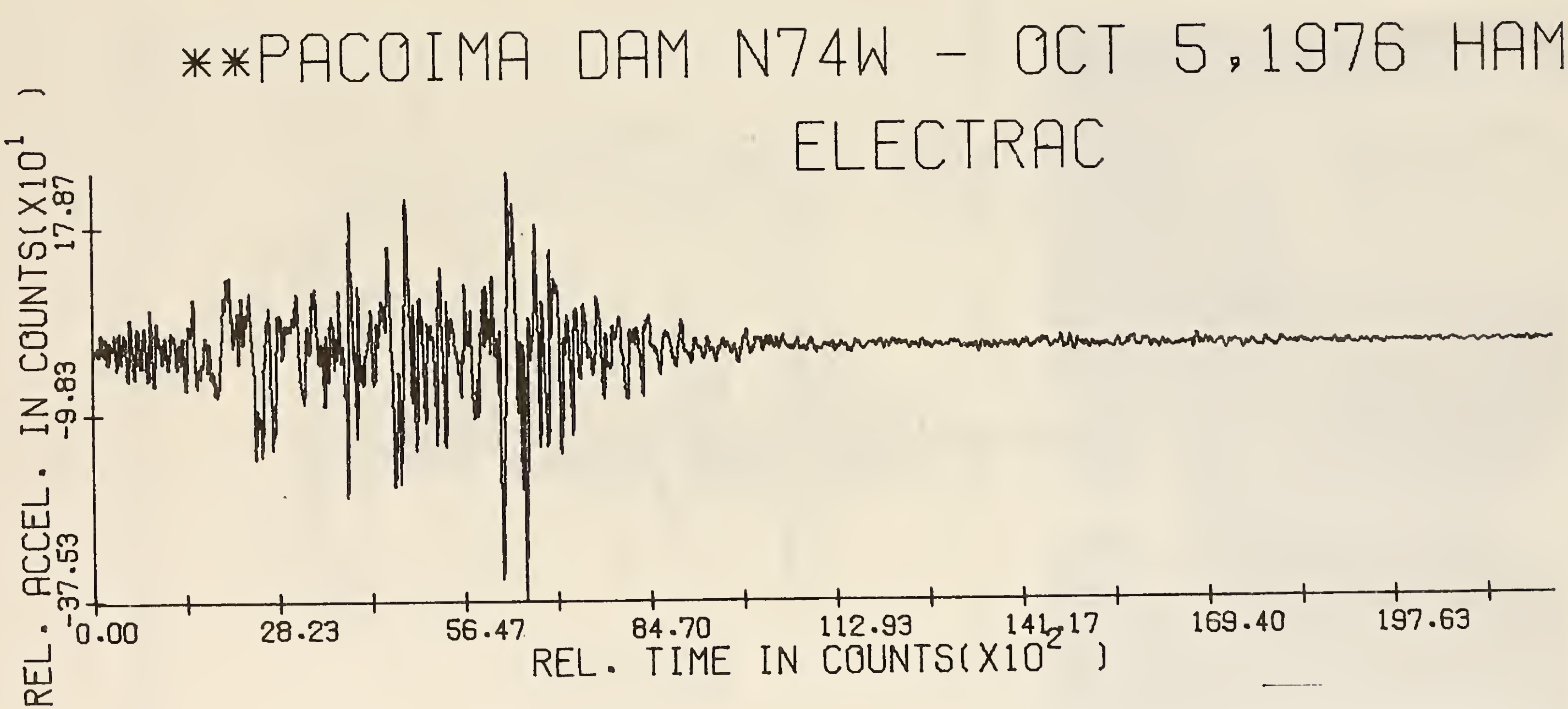


FIGURE 28 BASELINE CORRECTED ACCELEROGRAM FOR THE N76^{o}W COMPONENT, PACOIMA DAM, USING THE ELECTRAK DIGITIZER

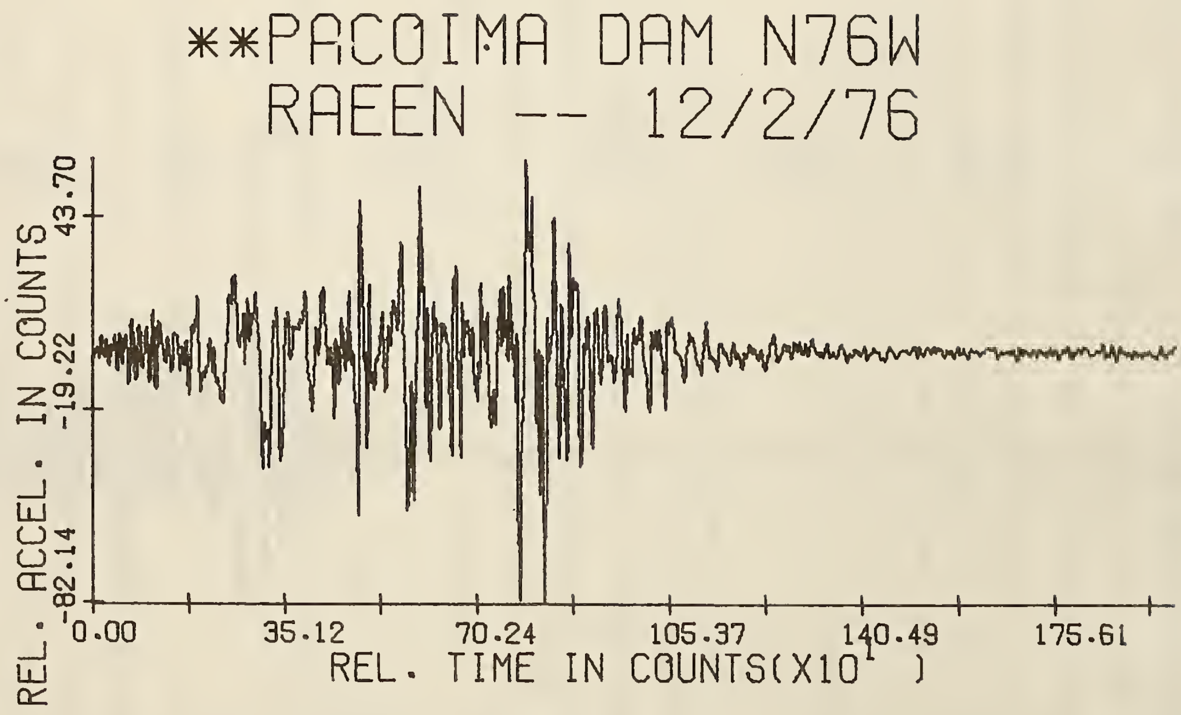


FIGURE 29 BASELINE CORRECTED ACCELEROGRAM FOR THE N76°W COMPONENT, PACOIMA DAM, USING THE BENSON-LEHNER DIGITIZER

FIGURE 30 MODEL 1010A PERKIN-ELMER MICRODENSITOMETER

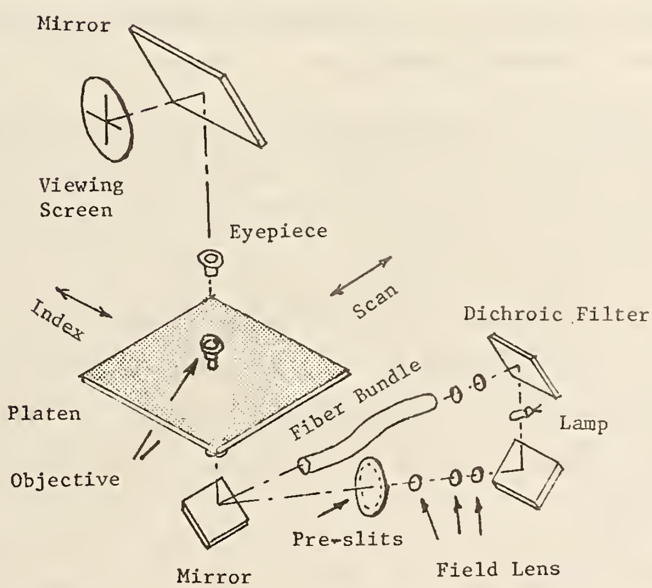


FIGURE 31 OPTICS OF PERKIN-ELMER MICRODENSITOMETER

operational at the time this research was prepared, the authors used an operational system at the Kitt Peak National Observatory for a 20-minute scan, which was all the time allowed under a crowded operation schedule at Kitt Peak. However, this was enough to establish the possibility of the system for accelerogram digitizing purposes and the problems that occur for this use.

The microdensitometer system is designed to take accurate readings of very small areas of a photographic plate or film at precise locations. Plates or film can be measured in terms of either density or transmission over a dynamic range in density of 0.0 to 5.115. Signals describing the measured density or transmission values are produced from a photomultiplier tube and amplifier, fed to an analog-to-digital converter and then, as digitized information, routed to the core memory of the controlling computer system. The end product is either output onto a magnetic tape, a strip chart recording, or a teletype page.

A Digital Coordinate Readout System (not shown) monitors and displays the X and Y stage positions in microns,* provides signal interrupts to the computer when scan limits are reached and also at setable intervals for density measurement. The model 1010A microdensitometer unit consists essentially of three systems: one to measure density or transmission information, one to show the storage in either or both X and Y directions, and one to generate precise storage position information.

Density measurement, Figure 31, is accomplished by passing a beam of light through the illuminating, lower,

---

* A micron for this machine is depicted to be 10^{-6} meters or 10^{-4} centimeters.

optical system, through the sample being tested, and on to the analyzing, upper, optical system which is symetrical to the first, then on to the photomultiplier tube. The illuminating optical path contains eight preslit apertures selected to suit a matching set of eight scanning apertures in the upper optical path. Four of the apertures are square and four are rectangular. The availability of two objective magnifications (X10 and X4) and four secondary magnifications provide a wide variety of apertures to choose from.

With only one run available, the authors decided to use a 40 x 40 micron slit size. This allowed a grid of 635 counts per inch in either the X or Y directions.

The photomultiplier converts the light intensity into a voltage signal, which is amplified by a logarithmic converter into density output values, i.e. from 0.0 to 0.115.

Each stage axis is driven independently by a precision DC servo motor/tachometer system ensuring uniformity of stage motion. The stage accommodates films or plates up to 10" x 10". For that reason a film negative with a maximum dimension of 9" along the record was prepared from the original contact opaqued negative. Also the middle (Down) acceleration portion was completely eliminated by opaqueing on the reduced negative. However the middle fixed trace was retained, while the fixed trace for the N76^{o}W and S16^{o}E components were eliminated. The software computer program did not allow differencing between the fixed trace and acceleration curve for the same component. The film was placed on the platen of the microdensitometer (Figure 31), and overlain with a clear glass plate to keep it flat.

14. Scan Parameters

A <u>scan</u> or <u>scan line</u> is defined as a single sweep (traverse) across the plate, the length of the sweep and the

distance between data points being specified by the user. When magnetic tape is used, this scan line will be written as one or more magnetic tape records, depending on the number of data points in the scan line. A scan pattern consists of a series of one or more parallel scan lines, with a specified distance between each scan line.

When magnetic tape is used, one complete scan pattern is written as one file on the magnetic tape. Therefore, the scan pattern covers a rectangle whose width is the length of one scan line and whose length is defined by the number of scan lines in the pattern and the distance between scan lines. The corresponding magnetic tape file will consist of NS (no. of scan lines) by NR (no. of points per scan) followed by an end of file. Two end of files signify end of information. During the 20 minute scan NS was equal to 1168, and NR was equal to 2850; thus about 1/4 of the total accelerogram was recorded. The data was placed on the tape in a raster scan pattern. A raster scan pattern is one where consecutive scans are done in opposite directions, while an edge scan pattern is one in which scanning is only done in one direction, with no data being taken on the return movement of the platen.

15. Reduction of Data - Microdensitometer

A software package OPSCAN * was used to retrieve the data from the tape in usable form. Essentially OPSCAN first unpacks the Identification Record at the beginning of each file, such as NS, NR, origin position etc. Then it unpacks each scan line. On the raster scan it returns the data in reversed order. As each scan is read the program searches the density values to be below a fixed limit (clear film),

---

* Obtained from the Solar Division, Kitt Peak National Observatory, and revised for use in digitizing.

counts the number of points in an array of points while crossing a clear Y width of film, and chooses the midpoint of that array to be output as a Y coordinate. The values of the X and Y coordinates are given as a position number of the midpoint in the scan pattern, or are relative representations of time (X) and acceleration (Y).

Figure 32 shows the data obtained for the first 200 scan lines, which cost approximately $5 to run on the CDC 6400 computer. For an 8" length of record the computer cost alone would have been approximately $127, not a small amount even though three components of acceleration can be digitized this way at one time by first creating a negative with three components and one fixed trace. At 635 counts/inch in both the X and Y directions there would have been 2850 x 635 x 8 = 14,478,000 data points to be examined in the full test record. Furthermore commercial rates for the required 4 hour scan on the microdensitometer (quoted at $60/hour by one microdensitometer installation in California) would add another $240 to the bill. This method is therefore quite expensive per run even when one has free access to the microdensitometer.

Figures 33 and 34 show the plots of Figure 32 onto Figures 16 and 20 respectively. Photographic problems on the accelerograph record prevent an accurate digitization representative of the acceleration curve. The automatic scanning system can select the midpoint of an array of points in its line of trace, but cannot discern where traces overlap. One way of overcoming this problem would be to make a negative of the original negative copy (that is acceleration curves would be black on clear film), and then cut separations between adjoining peaks and valleys using a film cutter, as indicated in Figure 35 but the separations are difficult to place accurately even with the aid of a magnifying glass. In view of the above limitations the microdensitometer method is not recommended for general accelerogram digitizing, but can be used where

PACOIMA SCAN1

	N74°W	Down Fixed Trace	S16°E		N74°W	Down Fixed Trace	S16°E
1	680	1597	2056	51	680	1590	2060
2	680	1597	2056	52	690	1590	2065
3	681	1596	2057	53	698	1590	2030
4	681	1597	2057	54	692	1589	2060
5	680	1597	2057	55	670	1588	2053
6	680	1597	2057	56	642	1588	2036
7	680	1597	2057	57	630	1588	2057
8	680	1598	2057	58	638	1588	2057
9	680	1597	2057	59	663	1588	2027
10	680	1597	2057	60	686	1588	2048
11	680	1597	2056	61	688	1590	2023
12	680	1598	2057	62	693	1590	2023
13	680	1598	2057	63	695	1590	2047
14	680	1598	2057	64	689	1590	2012
15	680	1598	2057	65	666	1590	2012
16	680	1599	2057	66	648	1588	2058
17	680	1598	2057	67	644	1589	2046
18	680	1599	2056	68	662	1590	2060
19	680	1600	2056	69	670	1590	2060
20	681	1600	2056	70	722	1590	2084
21	680	1600	2056	71	723	1588	2083
22	680	1600	2057	72	707	1588	2072
23	680	1600	2056	73	685	1588	2072
24	680	1599	2056	74	666	1588	2053
25	680	1598	2057	75	659	1589	2054
26	680	1598	2057	76	663	1589	2060
27	680	1598	2057	77	671	1589	2045
28	680	1599	2057	78	676	1588	2065
29	680	1598	2057	79	677	1588	2060
30	680	1598	2057	80	652	1588	2060
31	680	1598	2057	81	649	1588	2068
32	680	1598	2056	82	634	1588	2078
33	680	1598	2057	83	639	1588	2092
34	680	1596	2057	84	668	1588	2092
35	680	1597	2057	85	701	1588	1997
36	680	1597	2056	86	731	1588	2039
37	680	1597	2056	87	741	1588	2011
38	680	1597	2056	88	699	1588	2011
39	680	1597	2056	89	682	1588	2017
40	680	1597	2056	90	641	1588	2104
41	680	1597	2056	91	627	1588	2130
42	680	1597	2056	92	633	1588	2129
43	702	1588	2039	93	652	1588	2109
44	699	1588	2040	94	684	1588	2096
45	693	1588	2055	95	713	1588	2072
46	686	1589	2066	96	718	1588	2052
47	681	1589	2070	97	703	1588	2030
48	677	1589	2068	98	678	1588	2026
49	673	1590	2060	99	658	1588	2014
50	670	1590	2049	100	640	1588	2017

FIGURE 32 RELATIVE TIME-ACCELERATION COORDINATES X - Y FOR THE FIRST 200 LINES OF SCAN, MICRODENSITOMETER METHOD, PACOIMA DAM RECORD

101	628	1588	2031
102	629	1588	2059
103	635	1588	2072
104	648	1588	2089
105	685	1588	2089
106	725	1588	2089
107	762	1588	2089
108	778	1588	2067
109	759	1588	2064
110	703	1588	2064
111	633	1588	2057
112	583	1588	2057
113	576	1588	2051
114	593	1588	2054
115	622	1588	2063
116	644	1588	2063
117	671	1588	2063
118	711	1588	2086
119	750	1588	2077
120	769	1588	2036
121	767	1587	2007
122	698	1587	2007
123	644	1587	2007
124	606	1587	2007
125	611	1588	2070
126	601	1588	2068
127	602	1588	2068
128	610	1588	2068
129	646	1588	2066
130	661	1588	2066
131	708	1588	2066
132	734	1588	2080
133	735	1588	2032
134	720	1588	1999
135	663	1588	1999
136	657	1588	2012
137	638	1588	2028
138	619	1588	2056
139	606	1588	2066
140	604	1588	2066
141	620	1585	2066
142	619	1585	2066
143	693	1585	2066
144	762	1588	2089
145	782	1588	2078
146	781	1588	2068
147	762	1588	2074
148	708	1588	2074
149	608	1588	2074
150	541	1588	2074
151	546	1588	2074
152	554	1588	2074
153	582	1588	2074
154	668	1588	2074
155	760	1588	2024
156	791	1588	2016
157	793	1588	2022
158	790	1589	2040
159	712	1589	2040
160	647	1589	2040
161	628	1589	2040
162	595	1589	2040
163	586	1588	2096
164	600	1587	2081
165	600	1587	2081
166	650	1588	2057
167	655	1589	2051
168	656	1588	2055
169	661	1588	2036
170	668	1588	2077
171	676	1588	2077
172	692	1588	2079
173	713	1588	2079
174	720	1588	2071
175	710	1589	2069
176	681	1589	2069
177	654	1589	2091
178	641	1589	2105
179	643	1588	2099
180	655	1588	2087
181	677	1588	2070
182	700	1589	2062
183	725	1589	2064
184	740	1589	2069
185	743	1588	2070
186	734	1589	2068
187	707	1588	2069
188	671	1588	2077
189	640	1588	2081
190	625	1588	2090
191	623	1588	2082
192	628	1588	2065
193	631	1587	2059
194	631	1588	2040
195	634	1588	2030
196	641	1588	2026
197	656	1588	2027
198	681	1588	2052
199	718	1588	2095
200	739	1588	2076

Figure 32 Cont.

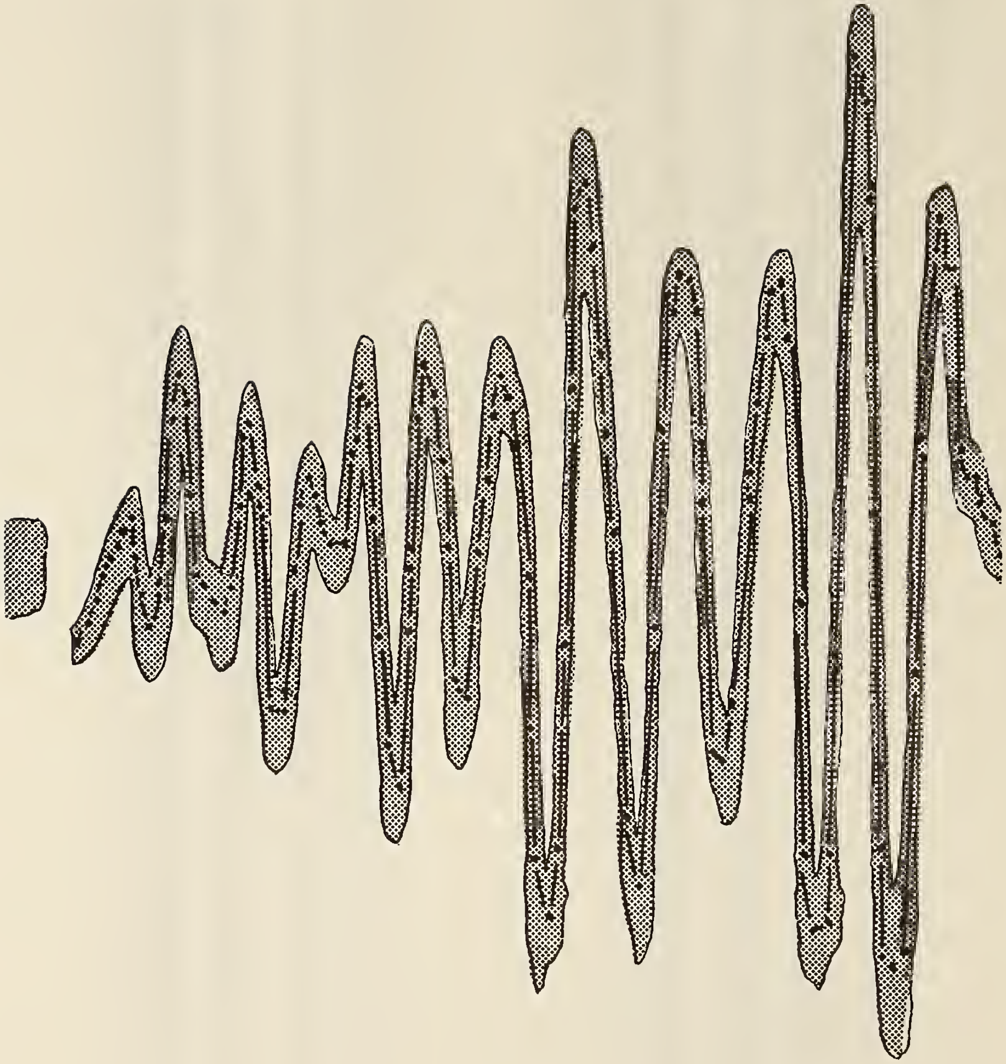


FIGURE 33 PLOT OF MICRODENSITOMETER DATA FOR 1.3 SECONDS OF RECORD, N74°W COMPONENT, PACOIMA DAM

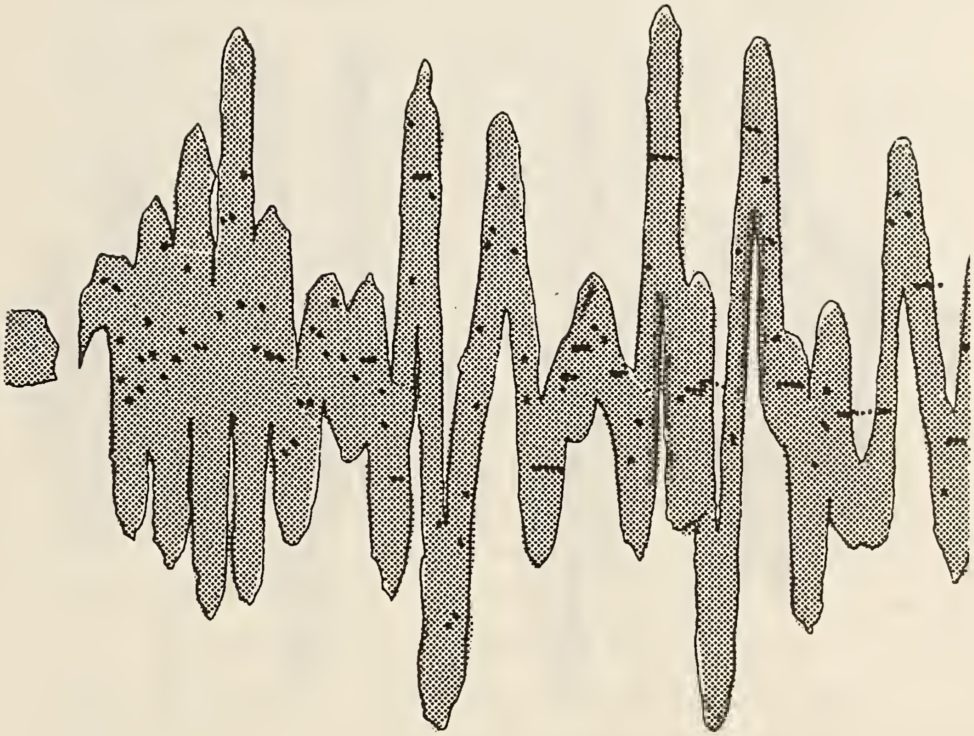


FIGURE 34 PLOT OF MICRODENSITOMETER DATA FOR 1.3 SECONDS OF RECORD, S16°E COMPONENT, PACOIMA DAM

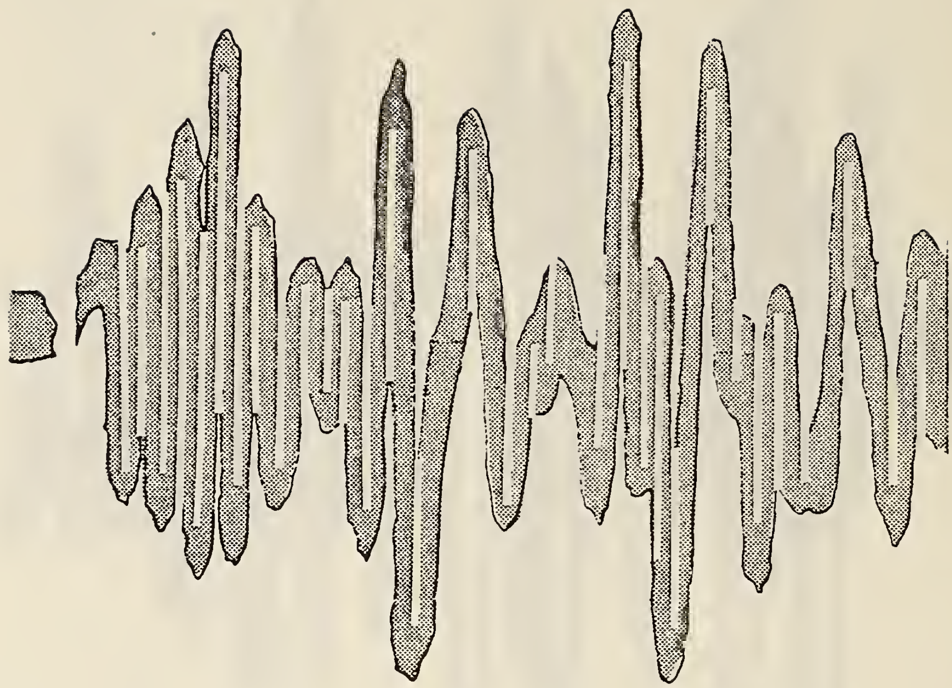


FIGURE 35 IMPROVED S16°E COMPONENT, PACOIMA DAM RECORD, OBTAINED BY CUTTING FILM EMULSION BETWEEN ADJACENT PEAKS AND VALLEYS OF CURVES

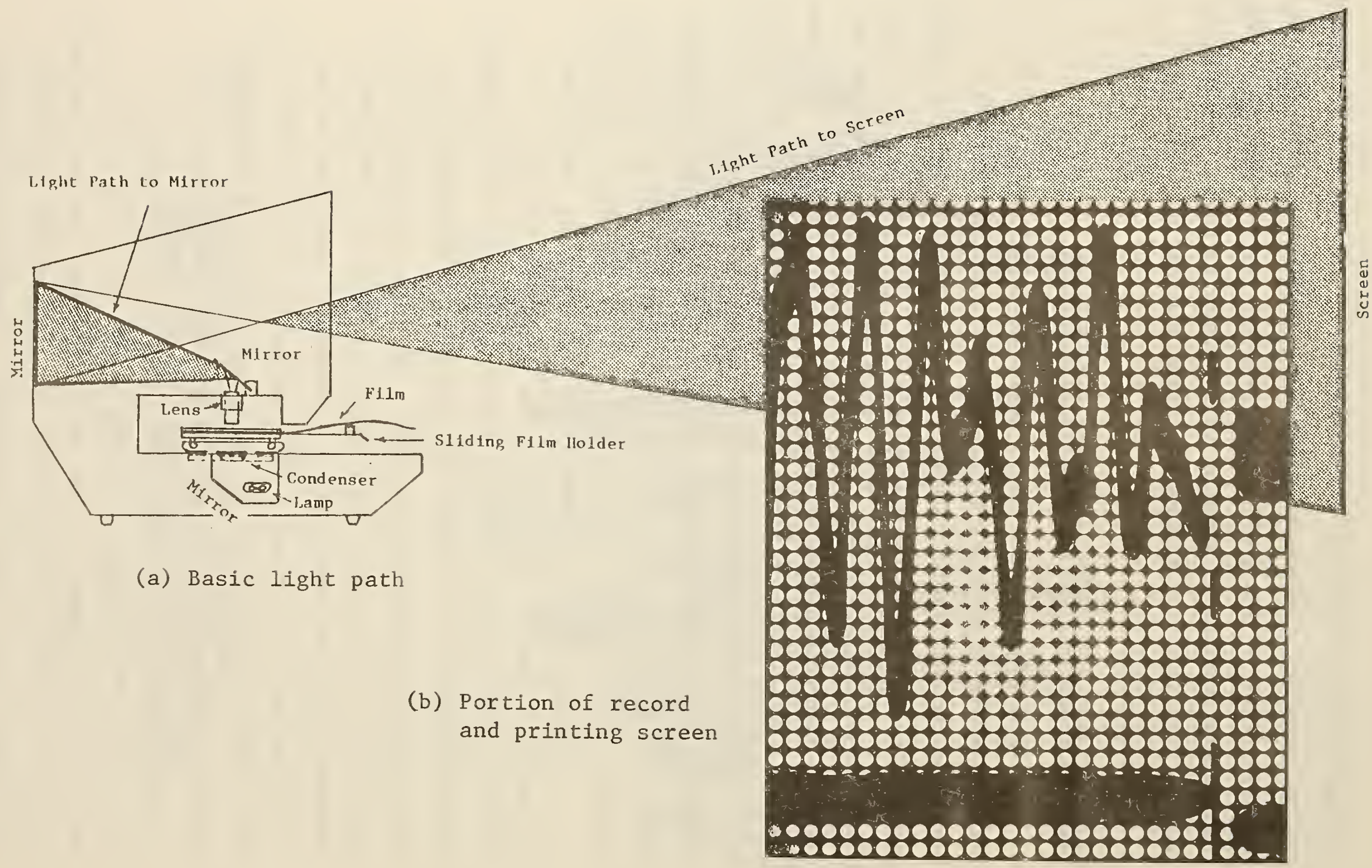
the peaks and valleys are far enough apart so that the automatic optical scanning system can produce a series of digitized points representative of the path of the original curve. For interested researchers wishing to examine this method further, the OPSCAN program that produced Figure 32, using the CDC 6400 computer, is listed in Appendix F .

16. Digitization with the Microfische Film Reader

Hand digitizing using a microfische film reader was first considered as an expedient measure when no commercial digitizer was available; however, the excellent results obtained might make this method highly desirable. The method takes more time than commercial digitizers, but results in less eyestrain, since no magnifiers are required. Also the problem of parallax, difficult to control on the Benson-Lehner machine, is eliminated.

One of the problems in any projection system is to establish linearity to scale for the projection. The problem was solved by projecting a linear grid together with the film record. The resolution can better that of the Electrak machine.

Figure 36 is a schematic of the setup for the microfische film reader. The front view screen is removed. A positive print of the original negative reduced to 8" maximum length (black on clear film) is placed in contact with 133 lines/inch halftone printing film. Halftone printing film is consistent in equal grid spacing, and serves as a background reference grid for establishing coordinate locations on the accelerogram. The printing film and positive accelerogram are then placed on the platen of the microfische film reader and projected onto a wall, on which a 36" x 36" paper grid with 1" c.c. heavy lines in both directions is already located. By suitable movement and focusing of the microfische film reader the enlargement of the printing screen can be arranged so that the center of the screen circles coincide as nearly as possible to all the 1" square grid crossings on the paper.



(a) Basic light path

(b) Portion of record
and printing screen

FIGURE 36 SCHEMATIC FOR PROJECTION OF MICROFISCHE FILM READER

In this way a linear projection of the accelerograph is focused on the wall at a greatly enlarged scale (the authors used a 24X enlargement lens). Figure 37 shows the resulting projection for the first 1.3 seconds of N76^{o}W Pacoima Dam record. An origin is established on the paper record and the digitized points are recorded. Since the platen will have to be moved several times to obtain the full record, different colored felt tip pens can be used for different sections of the record, projected onto the same grid paper. Afterwards the coordinate locations are merely read off of the paper grid and hand-punched onto cards. Figure 37 also shows an excellent plot of the scaled data. Some idea of the accuracy involved can be stated as follows:

The original full record was 28 seconds at 17 inches long. One inch on the Electrak machine $\frac{28}{17}$ = 1.647 seconds of record is identified with 1000 machine counts. The reduced record on the microfische is 8 inches long and has 133 lines of printing screen per inch or 16.625 screen lines/inch of original record. But, the record is projected so that a 1" square on the film becomes a 16.625" x 16.625" square on the wall. The paper grid on the wall is finely divided to 0.1 intervals and can be easily interpolated by eye to 0.01 intervals. Therefore, a microfische count of 1662.5/inch of original record is possible.

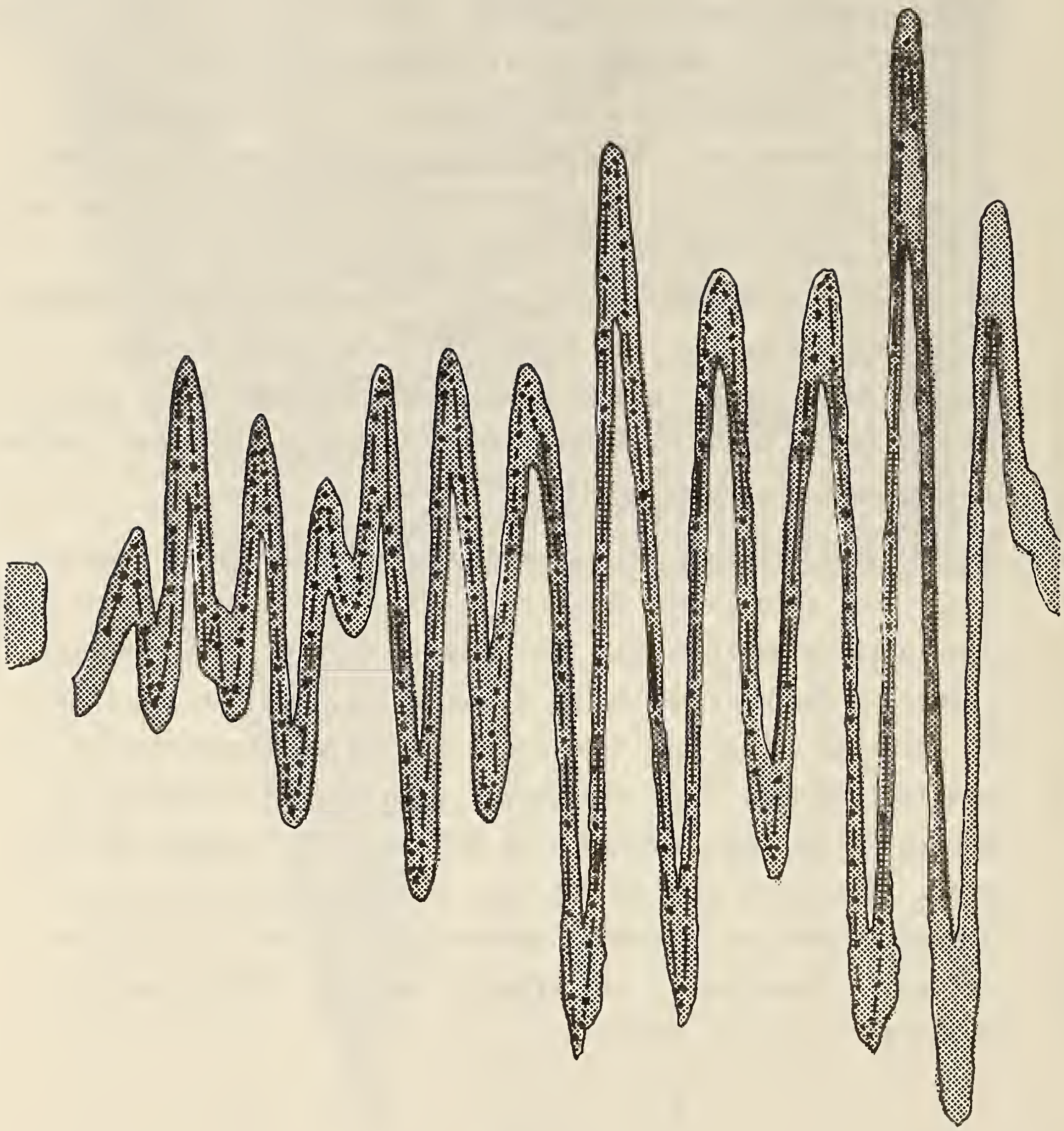


FIGURE 37 MICROFISCHE PROJECTION - FIRST 1.3 SECONDS OF N76°W PACOIMA DAM RECORD

III. PHASE1 SUBROUTINE - DIGITIZATION

1. Introduction

The PHASE1 subroutine is a preliminary computer program that processes raw digitized accelerogram data into a form that can be utilized by a second phase program (PHASE2 - Appendix D for filtering and integration. PHASE1 reads data that has been digitized from an Electrak 100 (on a 7 or 9-track magnetic tape) or a Benson-Lehner digitizing machine (on IBM key-punched computer cards). The program is written in FORTRAN IV and is designed for use on the CDC 6400 computer or IBM 360/370 O.S. computer. It has a capacity of 3000 digitized accelerogram points per component or about 30 seconds of an earthquake record.

PHASE1 is used as a "first step" data handling program to make necessary corrections prior to subsequent integration and can be used to punch corrected data on IBM computer cards. Any number of digitized components can be processed by PHASE1 as long as CP time is available (See User's instruction for details.)

2. Purpose

The quality of digitized accelerogram data depends highly on the expertise and skill of the operator. Since this human function is susceptible to error, a method of correcting these errors, before further processing, is of utmost importance. The primary purpose of PHASE1 is to locate accidental errors so that the user can make the necessary corrections before proceeding to subsequent filtering and integration of the accelerogram data.

In addition to human error, machine error can also nullify the quality of a good digitized record.

For either case, any integration and filtering scheme is only as good as the input data. The quality of the final result is directly related to the quality of the input data.

3. Features

PHASE1 produces a plot of the digitized data, through the use of subroutine PLOTTR. This is especially useful when the plot is reproduced to the exact dimensions of the original accelerogram trace. By placing the full-scale plot over the original accelerogram, deviations between the two are quite obvious if an error in the input data exists. This plot is a representation of the data that will be subsequently filtered and integrated; therefore, any errors must be corrected before further processing. (The actual correction procedure depends on the machine from which the data was originally digitized and is explained in detail in the corresponding digitizing section of this report.)

After these corrections (if necessary) are made, another full-scale plot may be made using the corrected data. If the new plot agrees with the original accelerogram, the user can now be assured that his data is correct and should yield good results when filtered and integrated.

Another basic feature of PHASE1 is to convert data recorded on a 7 or 9-track magnetic tape (by the Electrak machine) to punched cards. This is necessary before going on to the PHASE2 program. The Electrak machine recording errors are also corrected by PHASE1 (details of this portion are given in the TRAK 010 subroutine section in Appendix E.

As a final option of PHASE1 the data may be scaled and the baseline corrected. This feature yields acceleration values that are relative to a horizontal straight line which represents the average acceleration of the entire earthquake record. All acceleration values are thus given as being either positive or negative if they are greater or less than this average acceleration. Also the initial time value is changed to zero and all subsequent values of time increase accordingly. In effect, this feature of PHASE1 places the X origin at the first digitized point with the accelerogram trace oscillating about an average acceleration value (i.e., the X axis).

The options available to the user are sufficient to accomplish the necessary requirements before proceeding to further data processing. It should be noted, however, that if corrections are to be made before processing data in PHASE2, baseline corrected data cannot be used since the time and acceleration output values are not the same as the input data. Baseline corrected data should only be obtained after the user is assured that the data has been correctly digitized or that all necessary corrections have been made. When this requirement is satisfactorly fulfilled, the baseline correction option is exercised and the new data is ready for further processing in the PHASE2 subroutine.

4. How the PHASE1 Subroutine Works

PHASE1 consists of eight major steps which are as follows:

A. Pass 1 -- Plotting the digitized data to the scale of the original accelerogram.

   1. Read and write the title and main control information.

   2. Read in the raw digitized data, count the number of data points and write.
      Call subroutine PLOTTR to plot the data

and go to step 7.

7. Punch the data.

B. Pass 2 -- Prepare baseline corrected data after all corrections have been made to the raw data.

   1. Read and write the title and main control information.
   2. Read in the corrected raw data, count the number of data points, and write.
   3. Check the time values for continuously increasing times.
   4. Read and write the fixed trace data (if any) and deduct from the acceleration values.
   5. Adjust the data for initial time equal to zero and a new zero baseline.
   6. Plot the baseline corrected data.
   7. Punch the baseline corrected data for use in PHASE2.
   8. Return to step one and repeat the process if other raw corrected data is available.

<u>Step 1</u>

This step consists of reading and writing the title and the main control information input on the first four cards of the data. The main control information (second and fourth cards) direct the program to yield the desired output (i.e., a plot of raw or baseline corrected data and a punched deck of raw or baseline corrected data).

<u>Step 2</u>

The second step reads the raw digitized time-acceleration data from magnetic tape or raw or corrected digitized time-acceleration data from punched cards depending upon the value of "INPUTP" given in the fourth control card.* If the data is to be read from a magnetic tape, the data is first corrected and placed in a format (4x,2F6.0), usable by the

---

* See listing in Appendix D, page D-3.

computer. If the data is to be read from punched cards, the data is read in format (2F10.0) and consecutively placed in the TIME and ACCEL arrays. The number of data points are counted, checked to see that they don't exceed the size of the arrays, and then written on the output file.

Step 3

This step is included merely as a means of assuring that the data is continuously increasing with time. Since some digitizing machines are extremely sensitive, the possibility exists where a subsequent data point may be digitized with a time value less than the previous point. This possibility does not actually represent the intent and, therefore, should be eliminated. If a subsequent time value does happen to be smaller than the previous time value, the subsequent value is equated to the previous (greater) value.

Step 4

In this step, fixed trace values (if any are included), are read into memory, smoothed, and deducted from the acceleration values. This process eliminates errors that are caused by improper accelerogram alignment on the digitizer or on the accelerograph itself. The smoothing process simply applies a weighted mean (1/4, 1/2, 1/4) to each Y coordinate of the fixed trace and the two adjacent Y coordinates. The deduction process requires interpolating the smoothed Y coordinates to time values that coincide with each acceleration value and then deducting the corresponding fixed trace value from the acceleration value. For the above smoothing process it is recommended that fixed trace digitized points be equally spaced in time, say at intervals of the order of one-half second.

Step 5

This portion scales the time and acceleration values, subtracts the initial time from all subsequent time values and changes the acceleration values relative to an average acceleration. The time and acceleration values are scaled according

to the factors SCALET and SCALEA respectively. The acceleration trace is then integrated to obtain the "area" (under the trace). This area is divided by the total time of the record to yield an average acceleration value. The new baseline corrected acceleration is then the difference between the old acceleration and the new average acceleration value. Physically this sets the change in ground velocity from beginning to end of record to zero. When no fixed trace is available, the baseline is not only first translated to make the mean zero as above, but then a very small rotation is introduced to make the sum of the squares of the deviations from the zero line a minimum.

Step 6

The time-acceleration data is plotted in this portion of PHASE1. For pass 1 the call to subroutine PLOTTR provides the means of plotting the data to the exact size of the original accelerogram. No labeled coordinate axes are provided. The location of coordinate input data is described in the section on digitizing. For pass 2 it is necessary to first obtain the maximum and minimum values of the corrected data, and utilizing these values for the desired dimensions of the plot, the scaling factors are obtained. These scaling factors convert time-acceleration values to dimensions compatible for the plotter, and labeled relative coordinate axes are provided. In addition, the title of the data is printed on the output plot for both passes.

Step 7

The desired time-acceleration data is punched on computer cards in the format 4(2F10.3). The "4" yields four data points per card, each data point consisting of two quantities, i.e., the time and acceleration values.

Step 8

In this step control is directed to the beginning of the program where more data is read. If a title card is read,

the program expects to process another accelerogram component. If an end-of-file card is read the program terminates.

A flow chart of the PHASE1 subroutine is shown in Figure 38. Figures 28 and 29 show baseline corrected plots from Electrak and Benson-Lehner data respectively.

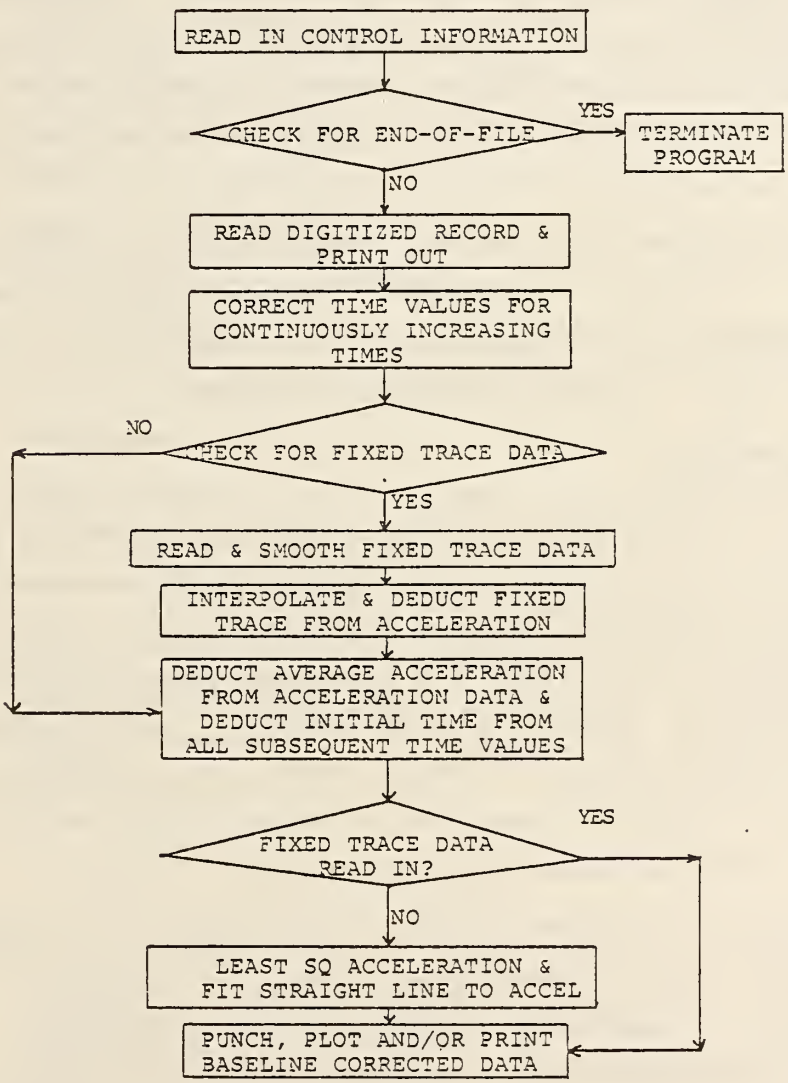


FIGURE 38 FLOWCHART OF PHASE1 SUBROUTINE

IV. INTEGRATION

1. Introduction

After the analog data has been digitized and corrected for accidental digitizing errors, it must be integrated to be of use in a design problem. The approach used by Cal. Tech. is basically a step by step process of integrating, least-squaring, filtering, and instrument correction. Integration of acceleration is necessary to obtain velocity and displacement. Filtering of the data is required to remove random and systematic errors that result from digitization and integration.

In this section, we shall discuss sources of errors, methods of integration, filtering, leastsquaring, and instrument correction, and how the Cal. Tech. program uses these methods to try to reduce the effects of errors.

2. Sources of Errors

One problem with obtaining actual true ground displacement from an earthquake accelerogram is the presence of extremely low and high frequency digitized errors. It has been found that these errors are sometimes random and sometimes systematic and are due to the nature of the recording and digitizing methods.[11] These errors contain long and short period components which do not actually exist in the original accelerogram and therefore must be eliminated.

Some of the causes of long period errors are: transverse play of the recording paper, warping of records, enlargement of records, imperfections in the digitizing machine, and imperfections in the recording instrument itself.

Short period (high frequency) errors are caused by imperfections in the recording instrument, random digitization errors, and inadequate resolution of the digitizing machine.

These errors are inherent in the present methods used to obtain actual ground displacement. To isolate and retain valid digital records of the accelerogram within a realistic frequency range, a low-pass digital filtering scheme is used. (This method is described later under "Methods of Filtering"). A low-pass filter "filters out" data containing high frequency recording errors and allows the lower frequency (long period) component to pass. Unrealistic low frequency errors are eliminated by first isolating the long period component and then deducting it from the original data. This is a way of high-pass filtering the data.

It has been found through empirical studies[11,12] that the limiting values of reliably retrievable digitized data are between 0.07 Hz and 25 Hz. Thus if any (sinusoidal-like) periods exist in the data longer than 16 seconds or shorter than 0.04 seconds ($T = \frac{1}{f}$), these periods will be eliminated.

Another source of error that any data reduction scheme must deal with is that of correlating actual ground displacement with instrument transducer response (See Figure 2-b). The relative movement x of the transducer is defined by the equation:

$$\ddot{x} + 2\omega_o \delta_o \dot{x} + \omega_o^2 x = -a$$

where δ_o is the critical damping coefficient of the recording transducer, ω_o is the natural frequency, and "a" represents the absolute ground displacement. Since high frequency errors are magnified by differentiation, the acceleration is low-pass filtered before the instrument correction is applied.

A third major source of error associated with calculating ground displacement from acceleration is the location of a zero baseline to represent zero acceleration. Since the accelerogram trace is triggered by the earthquake itself, the initial portion of the record is lost, denying the seimologist of the location of initial zero acceleration. To overcome this handicap, the digitized acceleration values are leastsquare fitted so that the sum of the squares of the differences between the zero baseline and the acceleration values is a minimum. (The mathematics of this procedure is presented in Appendix A.) According to reference 12, integration of the leastsquared acceleration magnifies long period errors in velocity and displacement; therefore, these quantities are high-pass filtered to eliminate the low frequences. It will also be shown that the leastsquaring procedure disturbs the record by its own mathematical routine.

3. Integration

The digitized data representing an accelerogram trace (See Figure A-1) is in the form of a series of coordinates defining a distinct location on a two-dimensional plane. The acceleration function is thus considered piecewise continuous consisting of many short straight-line segments (i.e., the dashed line). To integrate a function such as this, the "Trapezoidal Rule" (a numerical method for integrating a continuous function) is utilized. The Trapezoidal Rule is defined as follows:

$$V_i = \frac{\Delta t}{2} (a_i + a_{i-1})$$

where V is the integrated quantity of the function a and Δt is the interval along the time axis between corresponding consecutive values of a_i. Essentially, the function V represents the area under the acceleration curve between the time interval $t_i \rightarrow t_{i+1}$. The formula is an extension of the standard method for determining the area of a trapezoid.

Because of the nature of the input data (i.e., piecewise continuous), this method of numerical integration yields accurate results.

4. Methods of Filtering

Direct integration of accelerogram data from the digitized records does not accurately represent actual ground velocity and displacement. There are several reasons for this. Paramount among these is the fact that the recording instrument itself does not have the capability of recording frequencies higher than 25 Hz. Therefore, any data recorded at frequencies greater that 25 Hz are due to extraneous noise and do not represent accelerations caused by the earthquake.

Also it has been found [13] that present digitization methods introduce extremely low frequency (long period) random and systematic errors. Double integration of acceleration curves tends to magnify small errors and yield large, unrealistic displacement amplitudes. There errors are insignificant up to periods of about 16 seconds, but for longer periods, the errors become quite serious and distort the actual resulting ground displacement. The lower limit of accurately retrieved data (by the methods described in reference 13) is about 0.07 Hz.

Any resulting data outside the range of 25 Hz and 0.07 Hz is considered to be contaminated with erroneous noise and should be filtered out of the input data.

For low-pass filtering, we use an Ormsby [14] numerical filter to eliminate higher frequencies and "pass" data which contain frequencies below a certain rolloff frequency (See Figure B-1). The transfer function of filter weights is unity for all data corresponding to a frequency between zero and ω_c and attenuates rapidly to zero for data corresponding to a higher frequency. (The mathematical proof of the low-pass filter scheme is detailed in Appendix B.) This allows low frequency data (below ω_c) to pass unaltered and reduces high frequency data to zero.

To filter out long period components from the input accelerogram, the data is first smoothed by a running mean filter of length 0.36 seconds.[15] This removes high frequency (short period) components and leaves behind components in the data which have a lower frequency.

The remaining data is further relieved of higher frequencies by filtering with the Ormsby low-pass digital filter. Briefly stated, the Ormsby low-pass filter scheme works as follows: the time varying input data A(t) (acceleration as a function of time) is multiplied by certain weights (real numbers) such that their product yields output data containing only low frequency components of the data. In a sense, this process "filters out" unwanted higher frequencies. The problem is to specify a weighting function h(t) to achieve this goal. The filtered input function can be represented by:

$$F(A(t)) = \sum_{i=-\infty}^{\infty} h_i A_{t-i}, \quad t = 0, \pm 1, \pm 2$$

where the h_i's are the filter weights. If the input time series is represented by $A(t)=e^{i\omega t}$, then the output filtered

series will be $H(\omega)e^{i\omega t}$, where $H(\omega)$ is the transfer function of the filter. From the above equation it follows that the transfer function of the filter F is

$$H(\omega) = \dot{h}_i \ e^{i\omega t}$$

The frequency transfer function $H(\omega)$ is specified to be unity in a frequency band where frequencies are allowed to pass unaltered and zero outside this frequency band. The weights associated with this desired transfer function are determined by:

$$h_i = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{i\omega t} \ H(\omega) \ d\omega$$

Evaluating the above expression for the filter weights (See Appendix B) and specifying a straight line attenuation beyond the desired frequency (ω_c), yields the formula

$$h_n = \frac{\cos 2\pi n\lambda c - \cos 2\pi n\lambda t}{2\lambda_r (\pi n)^2} \qquad \begin{array}{l} n = 0, \pm 1, \ldots, N \\ \lambda_t = \lambda_c - \lambda_r \end{array}$$

where λ_c and λ_r are normalized frequencies with respect to the frequency of the equally spaced time data. These weights are calculated in subroutine ORMSBY (loop #17) and applied to the equally spaced input data (AENTER).

The resulting output data is the long period component which remains after low-pass filtering. This long period component (caused by errors in recording and digitizing) is deducted from the original accelerogram thus "filtering out" the undesirable low frequencies.

V. PHASE2 SUBROUTINE - INTEGRATION

1. Introduction

Subroutine PHASE2 is a computer program that filters and integrates digitized accelerogram data to provide a final time history of corrected acceleration, velocity, and displacement. The processing of accelerogram data is similar to that adopted by the California Institute of Technology,[10] The input data is in the form of coordinate values representing baseline corrected digitized time and acceleration data of an earthquake record. PHASE2 processes this data by filtering out frequencies unrelated to the actual earthquake and integrating (using the Trapezoidal Rule) to obtain the corresponding velocity and displacement. The output is in the form of a listing of the above quantities and a plotted graph, if desired.

The program is written in Fortran IV and has a capacity of up to 54.95 seconds of record and 5500 interpolated points. This is ordinarily large enough to accommodate most accelerograms.

This portion of the report is intended to inform the reader of the purpose of PHASE2 and how the program carries out the data processing. Justification of the various functions are contained in other portions of this report and in the noted references.

2. Purpose

The primary function of the PHASE2 program is to filter and integrate accelerogram data. Filtering is necessary to remove various types of errors that result in the process of obtaining digitized data from an accelerogram trace.

The second primary function of PHASE2 is to integrate

the corrected acceleration values. This process produces ground velocity and displacement as a function of time. These quantities can then be used in the design spectra for determining appropriate stresses of a structure during a similar eqrthquake.

3. How the PHASE2 Subroutine Works

PHASE2 basically consists of seventeen major steps which are as follows:

1. Read the title and main control information.
2. Read the baseline corrected accelerogram data, count the number of digitized points, and write on the output file.
3. Scale the data and check the time values for continuity.
4. Interpolate the data to 0.01 seconds.
5. Low-pass filter the data to remove undesired high frequencies.
6. Remove alternate data points.
7. Correct for instrument response to obtain absolute ground acceleration.
8. Leastsquare the acceleration values and save.
9. Apply a running mean filter (Holoway) to smooth the data.
10. High-pass filter the decimated data from Step 9 to eliminate unwanted low frequencies and deduct from acceleration saved in Step 8.
11. Integrate the new acceleration to obtain velocity; leastsquare the velocity to obtain a correction term and deduct the correction term from the acceleration.
12. High-pass filter the velocity.
13. Integrate the new velocity to obtain the displacement.
14. High-pass filter the displacement.

15. Correct the acceleration values for new changes to the velocity.
16. List final acceleration, velocity and displacement time histories.
17. Plot final acceleration, velocity and displacement time histories.

Section 1 (See flowchart, Figure 39, at end of this section.)

The first section of PHASE2 reads the title of the baseline corrected time-acceleration data and the main control data. The title is kept in memory to be output at various times in the program. The main control data gives the program necessary information for applying an instrument correction, scaling the input data to units of seconds and gravity, and plotting control.

Section 2

This section reads the input data in Subroutine REDATA and writes it out so the user can check the listing to see that it agrees with the intended input. If an error exists at this point, all subsequent filtering and integration will be invalid. REDATA also checks the times to assure that all time values are increasing (or equal) and that the maximum difference between consecutive times does not exceed 1/4 of a second. This is included because data points further apart than 1/4 second are either inadvertantly omitted or such data produces erroneous output. If the data is further apart than 1/4 second, the information is truncated after this point and the program continues processing with the truncated data.

Section 3

The computations of this section are performed in the subroutine DATALT (for "data alter"). Here the time values are scaled to units of seconds and the acceleration

values are scaled to units of cm/sec^2. If the first input time is not zero, this function is accomplished by deducting the initial time value from all subsequent times, thus relating the beginning of the earthquake to a zero time origin.

Section 4

In this section, the acceleration values are interpolated to 0.01 seconds apart. This operation is carried out in subroutine EQLSPC (for "equal spacing"). EQLSPC utilizes a linear interpolation scheme of the form:

$$ACCEL_{i+0.01} = ACCEL + (TIME_{i+0.01} - TIME_i) \times$$

$$(ACCEL_{i+1} - ACCEL_i) / (TIME_{i+1} - TIME_i)$$

where i = 1,2, --- T representing equally spaced time values.

Section 5

At this point, the data is low-pass filtered, i.e., all frequencies greater that 25 Hz are eliminated. The subroutine ORMSBY (after Joseph Ormsby who first introduced the filter) does all the low-pass filtering throughout PHASE2. Details of this filtering scheme are given in Appendix B.

Section 7

If values are given for making an instrument correction because of damping in the recording system, the correction is carried out according to the formula:

$$\ddot{x} + 2\omega_o \delta_o \dot{x} + \omega_o^2 x = -a(t)$$

where: $\ddot{x}$ = the second derivative of acceleration and is computed in the program numerically; i.e.,

$$\ddot{x} = (x_{i-1} - 2x_i + x_{i+1}) / \Delta t^2$$

$\dot{x}$ = the first derivative of acceleration and is computed numerically; i.e.,

$$\dot{x} = (x_{i+1} - \dot{x}_{i-1})/2\Delta t$$

x = the acceleration (this term actually represents displacement of the transducer mirror which is a measure of acceleration). See Figure 2-b.

ω_o = natural frequency of the accelerometer transducer computed by,

$\omega_o = 2\pi/T$ (where T = natural period of transducer)

δ_o = percent of critical damping of seismometer

$a(t)$ = absolute ground acceleration.

Section 8

This portion of PHASE2 is accomplished by subroutine LESTSQ (Leastsquare). The function of leastsquaring is to place a straight line through coordinate data which best represents a function of all the values. (The mathematical theory of leastsquaring is detailed in Appendix A). In subroutine LESTSQ, the acceleration values are leastsquare fitted to a new baseline and these values are placed in a separate array to be called upon later in the program.

Section 9

Subroutine HOLWAY smoothes the acceleration values by utilizing a running mean filter. This is done by replacing the value of a point x_i with the average of itself and nine adjacent points on each side of x_i.

Section 10

High-pass filtering is accomplished by first decimating the data (i.e., using every 10th point) to acceleration values 0.2 seconds apart. The decimated data is then low-pass filtered to isolate the low frequencies, interpolated back to 0.02 second time intervals and deducted from the acceleration values saved in Section 8. By deducting low frequency data (not representing earthquake motion) from initial data, high-pass filtering is thus conveniently and simply carried out by utilizing only the low-pass filter.

Section 11

In this portion of PHASE2, the new acceleration values are integrated to obtain velocity, the velocity is least-squared and the leastsquare correction term ("B" in subroutine LESTSQ) is subtracted from the acceleration values to correct the acceleration for integration errors. This process is performed in the same LESTSQ subroutine by setting the parameter "NPASS" to 2, thus bypassing the leastsquaring operation until the acceleration has been integrated. Then the leastsquaring operation is performed to obtain the correction term.

Section 12

Here the velocity is high-pass filtered to remove the erroneous lower frequencies of velocity. This result is attained similar to Section 10 except that the velocity (instead of acceleration) is first low-pass filtered and then deducted from the previous velocity.

Section 13

In this section of PHASE2, the new velocity is integrated to obtain the corresponding displacement.

Section 14

As in Section 12, the displacement is high-pass filtered to eliminate erroneous low frequencies.

Section 15

The error introduced by integration (of velocity in Section 11) is subtracted from the acceleration in subroutine INTERP. Although subroutine INTERP is called later in the high-pass filtering operation, the error compensation is applied by means of the parameter "NPASS". When NPASS=1, the acceleration is corrected. In the second call to INTERP, there is no need to further correction acceleration, NPASS is set to 0, and the acceleration correction procedure is bypassed.

Section 16

This portion of PHASE2 lists the corrected time-acceleration values, time-velocity values, and time-displacement values. In each case, the values are given at intervals of 0.02 seconds. The listed arrays are output as follows: The time coordinates are given first and then the corresponding acceleration, velocity and displacement following immediately after.

Section 17

The final operation of the PHASE2 subroutine consists of plotting a coordinate graph of the final quantities of acceleration, velocity, and displacement. This is accomplished by subroutine PLTDAT (Plot data) and several Calcomp library subroutines. The plotted output size is controlled by the first control card. The three graphs plotted in this area represent the data listed in Section 16. In addition, the low-pass filter weights can be plotted.

A flow chart of the PHASE2 subroutine is shown in Figure 39.

ARRAY					STATEMENT NO.	COMMENT
ACCEL	TIME	VEL	DISP	LAST		
Input Acceleration Data-ACCEL1	Input Time Data-TIME			No. of Input Data Points		Read Time and Accel data -- REDATA -- Place in Time and Accel arrays.
ACCEL1 * SCALEA * 980.665	TIME * SCALET					Scale Time and Accel data -- DATALT -- Place in Time and Accel arrays.
			Unequal Scaled Time	Max. no. of .01 sec. intervals		Unequal spaced Time data stored in Disp array - PH2
ACCEL2	ACCEL2 TIME at 0.01 sec.					Interpolate Accel to get Accel2 at 0.01 sec -- EQLSPC -- uses unequally spaced time in Disp array.
ACCEL2		ACCEL2 Corrected at .01 sec.				If N5WAY = 1, or N6WAY = 1, call HORIZ. Accel2 corrected to make area under curve zero. Returned to Accel array after first placing in Vel array.
ACCEL4 at 0.02 sec.	ACCEL2 filtered or ACCEL3 BTIME at 0.02 sec.					If (CD.LE.0.0 or T.LE.0.0) go to 40, otherwise call Ormshy 25 - 27 cycles -- low pass. Filtered Accel stored in Time array. Then PH2 transfers the filtered Accel back to Accel array -- Accel4 at .02 sec. Create B-Time. Go to 90
ACCEL4 at 0.02 sec.	BTIME at 0.02 sec.				40	Decimate Accel2 to get Accel4 at .02 seconds. Create B-Time.
						If (CE.LE.0.0 or T.LE.0.0) go to 110. No instrument correction is made. Go to 90
ACCEL4	BTIME at 0.02 sec.		ACCEL4		90	Correct for instrument response. Accel first placed in Tinst (disp) then replaced in Accel array.

					110	If (NEWWAY EQ. 1) go to 130.
						If (N2WAY EQ. 1) go to 130.
						If (N3WAY EQ. 1) go to 130.
						If (N4WAY EQ. 1) go to 130
						If (N5WAY EQ. 1) go to 270
						If (N6WAY EQ. 1) go to 330
						Go to 150.

ACCEL4 Corrected		ACCEL4 Corrected			130	Call HORIZ. NPASS = 6. Correct Accel4 to make area under Accel4 curve zero. Return to Accel array after first placing in Vel array. Go to 170.
		ACCEL5 New ACCEL5	TEMP(VEL) New VEL		150	Call LESTSQ. NPASS = 1. Accel4 now leastsquared to get Accel5. Integrate Accel5 to get Vel (temp). Leastsquare Disp array to get velocity correction B, and subtract B from Accel5 to get new Accel5. Correct Vel = temp - correc.

FIGURE 39 FLOWCHART OF PHASE2 SUBROUTINE - CAL. TECH. VERSION

ARRAY					STATE-	COMMENT
ACCEL	TIME	VEL	DISP	LAST	MENT NO.	
ACCEL5					170	Accel5, replaced in Accel array. - PH2
						If (N3WAY EQ. 1) go to 270.
ACCEL6 (Holoway filtered)	ACCEL6(H. filtered) TIME at .02 sec.				190	Call HOLWAY. Places filtered Accel in Time array then replaces it into Accel array as Accel6.
ACCEL7 Decimated at .2 sec.	CTIME at 0.2 sec.					Accel data now decimated - PH2.
ACCEL8 at 0.2 sec.	ACCEL8 at 0.2 sec.		CTIME at 0.2 sec.			Call ORMSBY to filter Accel. Stores filtered Accel as Accel8 in Time array. PH2 replaces it in Accel array and stores CTIME at 0.2 sec. in Disp array.
ACCEL8 at 0.02 sec.	TIME at 0.02 sec.					EQLSPC interpolates Accel at .02 sec. Uses Time at .2 sec. in Disp array. Recreates Time at .02 sec.
ACCEL9 = ACCEL5 - ACCEL8					260	Subtract Accel8 from Accel5 to get Accel9 - PH2.

					270	If (N2WAY EQ. 1) write final Accel.
						If (N3WAY EQ. 1) write final Accel.
						If (N5WAY EQ. 1) write final Accel.
						If (N2WAY EQ. 1) go to 280
						If (N3WAY EQ. 1) go to 280
						If (N5WAY EQ. 1) go to 280
						Go to 290.

		Integrated VEL			280	Call HORIZ. NPASS =2. Integrate Accel to get Vel. Go to 300.
ACCEL9 Corrected for Velocity Change at 0.02 sec		VEL Corrected	VEL(TEMP)		290	Call LEASTSQ. NPASS = 2. Integrate Accel9 to get velocity (temp) in Disp array. Then leastsquare velocity and deduct correction term B from Accel9. Correct Vel = Temp - A - B * Time.

					300	If (N2WAY EQ. 1) go to 390
						If (N3WAY EQ. 1) go to 390
						If (N5WAY EQ. 1) go to 390

					330	Vel data now decimated to get Vel 1 at .2 sec. - PH2.
	ORMSBY Filtered VEL1 at 0.2 sec.					Call ORMSBY. Delt = .2. Filtered Vel placed in Time array.
ACCEL9 Corrected for Velocity change	ORMSBY Filtered VEL1 at 0.02 sec.					Ormsby filtered Vel placed in Vel 1 array at .2 sec. - PH2. Call INTERP. Vel 1 is returned through exit into Time array at .02 sec. Accel9 corrected for change in velocity due to filtering.

FIGURE 39 (Cont.)

ARRAY					STATEMENT NO.	COMMENT
ACCEL	TIME	VEL	DISP	LAST		
	TIME at 0.02 sec.	VEL = VEL - VEL1			390	Subtract filtered Vel from initial Vel to get new velocity - PH2. Write final velocity.
			Integrated DISP at 0.02 sec.			Integrate new Vel to get Displacement. If (N2WAY EQ. 1) N02 = N02 + 1

						If (N2WAY EQ. 1) write final Disp.
						If (N3WAY EQ. 1) write final Disp.
						If (N4WAY EQ. 1) write final Disp.
						If (N5WAY EQ. 1) write final Disp.
						If (N2WAY EQ. 1) go to 470.
						If (N3WAY EQ. 1) go to 470.
						If (N4WAY EQ. 1) go to 470.
						If (N5WAY EQ. 1) go to 470.
						Decimate Disp and place in Displ array at .2 sec. Pass to Ormsby.

	DISP1 at 0.2 sec.					Call ORMSBY. Delt = .2. Filtered Disp placed in Time array.
						Ormsby filtered Disp placed in Disp 1 array at .2 sec.
	DISP1 at 0.02 sec.					Call INTERP. Disp 1 is returned in Time array at .02 sec.
	TIME at 0.02 sec.		Final DISP = DISP1 - DISP at 0.02 sec.			Subtract Disp 1 from Disp. Recreate Time.
						If (N2WAY EQ. 1) go to 470 Write final acceleration.
ACCEL9 = ACCEL9/G Final						Correct Accel to units of gravity.

FIGURE 39 (Cont.)

VI. INTEGRATION - MATHEMATICAL FUNCTIONS

1. Introduction

How valid is the time-displacement curve as a result of the integration procedures of PHASE2? One way to study this is to input digitized time-acceleration data for mathematical functions with a known time-displacement history and compare the known history with the results of PHASE2.

2. Constant Acceleration Function

Figure 40-a shows a constant time-acceleration curve, extending from zero to 20 seconds, at a constant acceleration rate of 2 cm/sec^2. It was digitized every 1/15 second,

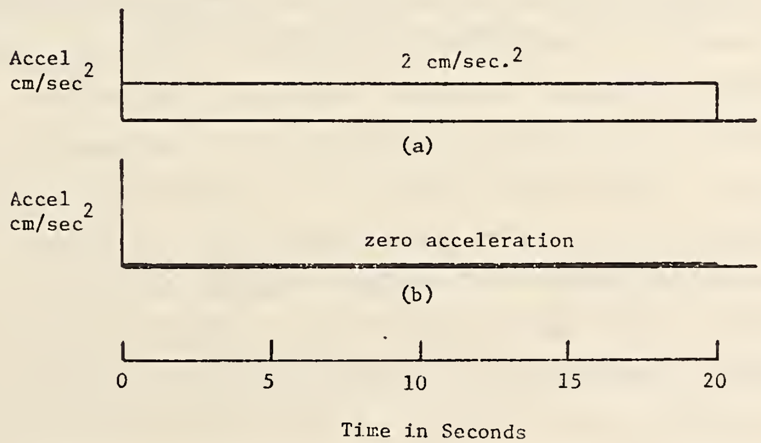


FIGURE 40 PROGRAM HISTORY - CONSTANT TIME-ACCELERATION CURVE - CAL TECH VERSION

resulting in 301 points, and processed by PHASE2. Write statements were inserted at various stages of the program so that the authors could follow the changes in the data. Follow the flow path in the program via Figure 39.

SCALET was given a value of 1, and SCALEA was given a value of 1/G. Therefore, through the REDATA and DATALT subroutines the data was interpolated at every 0.01 seconds resulting in 2001 points of constant value at 1.999 cm/sec^2. (Round off error truncated the results from 2.000 to 1.999 cm/sec^2.) Since the data originated from a mathematical function, ISHORT was given a value of 1 to bypass the instrument correction statements, there being no damping to consider.

Alternate data points were then eliminated in PHASE2, resulting in 1001 points of constant value at 1.999 cm/sec^2. The maximum time was still 20 seconds.

The program next called the LESTSQ subroutine to leastsquare the data. The leastsquare correction became a horizontal straight line at a value of 1.999 cm/sec^2. The correction line serves the function of a new baseline for the data. Differences between the acceleration value of the baseline and the acceleration input data to LESTSQ now became the new data, which in this case resulted in a zero value of acceleration for the entire record, as shown in Figure 40-b. Thereafter, no changes in acceleration occurred throughout the remainder of the program. Also the resulting velocity and displacement curves became zero throughout the record. Compare the results with the known integration for velocity and displacement by the mathematical procedures of calculus, Figure 41. Obviously there is an incompatibility between realistic results and the computer program results.

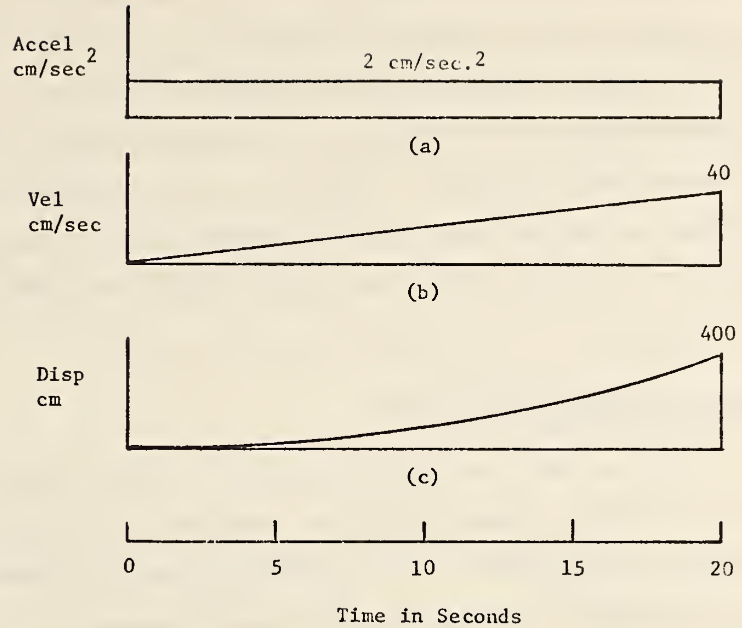


FIGURE 41 INTEGRATION OF CONSTANT TIME-ACCELERATION CURVE - CALCULUS PROCEDURE

We must be careful about the conclusions we draw from this first example. A constant acceleration curve is not an earthquake type of curve, since the final velocity never returns to zero.

The leastsquaring procedure, intended to recognize that initial and final velocities should be zero with earthquake data, fit the best average straight line to the input

data so that the sum of squares of the deviations from the new baseline to the curve are minimized. Even in this case had a horizontal baseline been placed so that the area under the acceleration curve was zero, the same computer results would have been obtained. Therefore, one conclusion we can draw from the above results is that *the program cannot correctly operate on data unless the initial and final velocities are zero.*

3. <u>Sawtooth Acceleration Curve</u>

Let us next consider a sawtooth acceleration curve, Figure 42, whose inital and final velocities do become zero by calculus integration. The time-acceleration curve was digitized at 1/15 second intervals, and processed through PHASE2. SCALET = 1, SCALEA = 1/G and ISHORT = 1. Therefore, the curve remained unchanged through the REDATA and DATALT subroutine. The data was interpolated in EQLSPC to 0.01 seconds, resulting in 2001 points and later cut to 1001 points in PHASE2 as every alternate point was discarded.

In the LESTSQ subroutine the area under the curve was calculated as zero (correct) and the displacement as 249.824 cm (round off from 250 cm). A = 3.7474, B = -.3747.

Figure 43 shows the changes in data at various stages in the program. Now the new baseline became a sloping straight line at a value of 3.7474 and a negative slope of -.3747. When the acceleration data is adjusted to the new baseline, we observe a startling change in the data and can draw an immediate conclusion: *Even if the initial and final velocities become zero, the leastsquaring method will distort the original data.* In some cases such as an earthquake record, with many cycles of alternate signs for the acceleration, the distortion

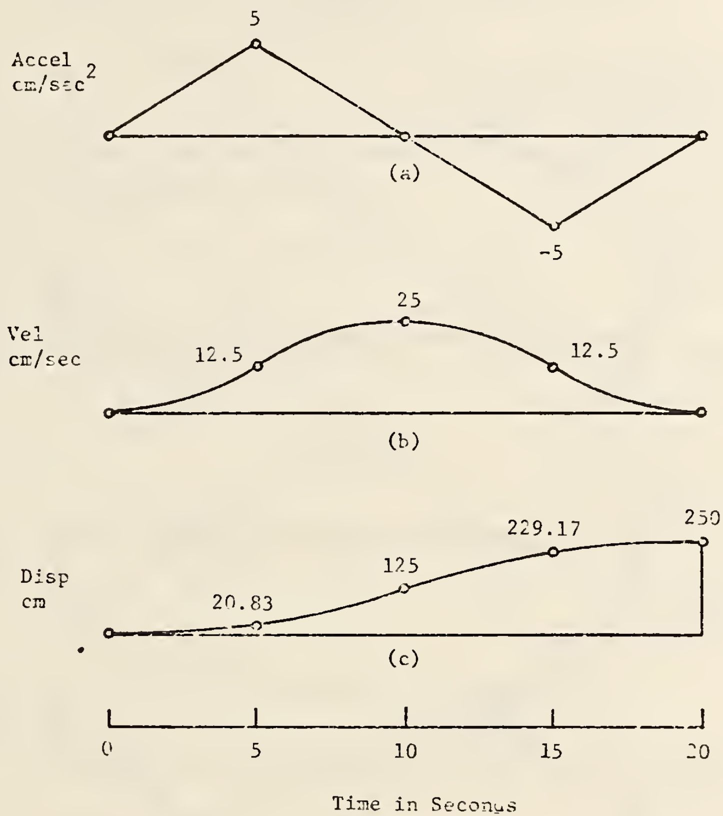


FIGURE 42 INTEGRATION OF SAWTOOTH TIME-ACCELERATION CURVE - CALCULUS PROCEDURE

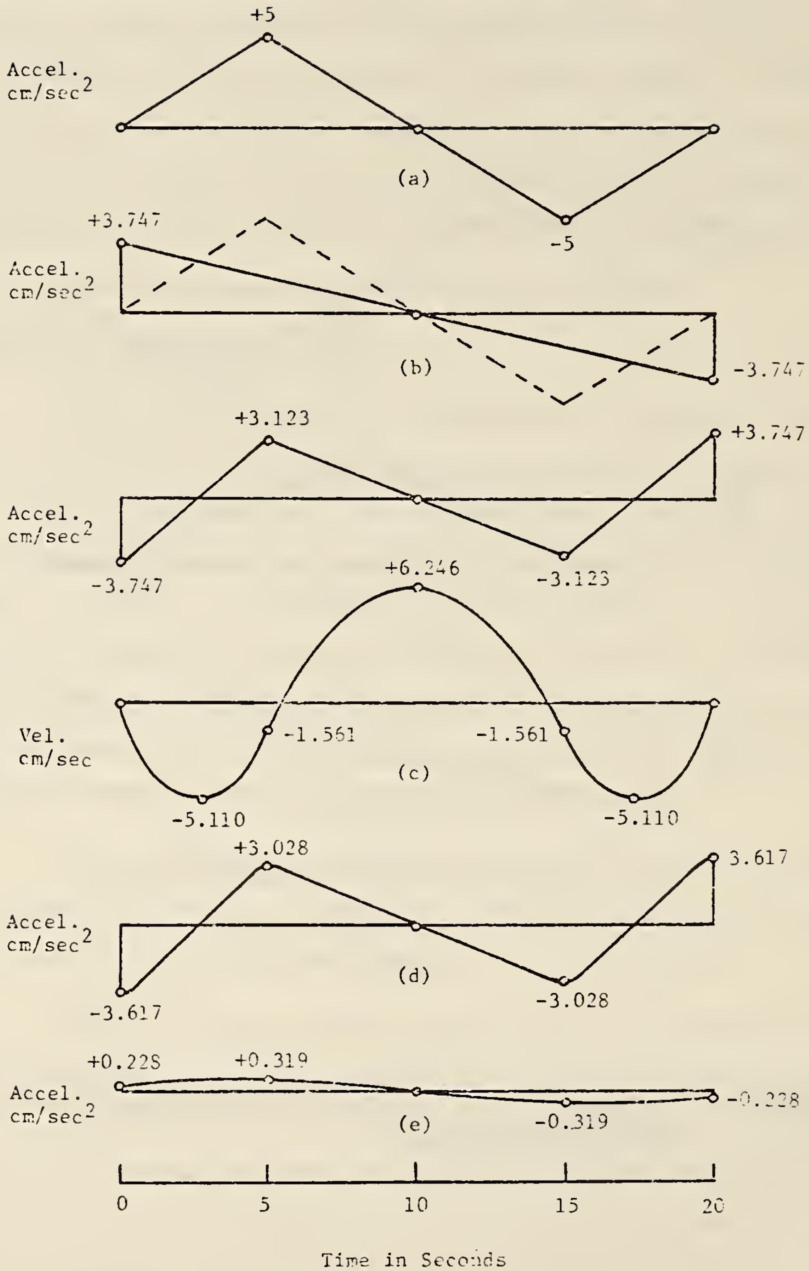


FIGURE 43 PROGRAM HISTORY - SAWTOOTH TIME-ACCELERATION CURVE - CAL TECH VERSION

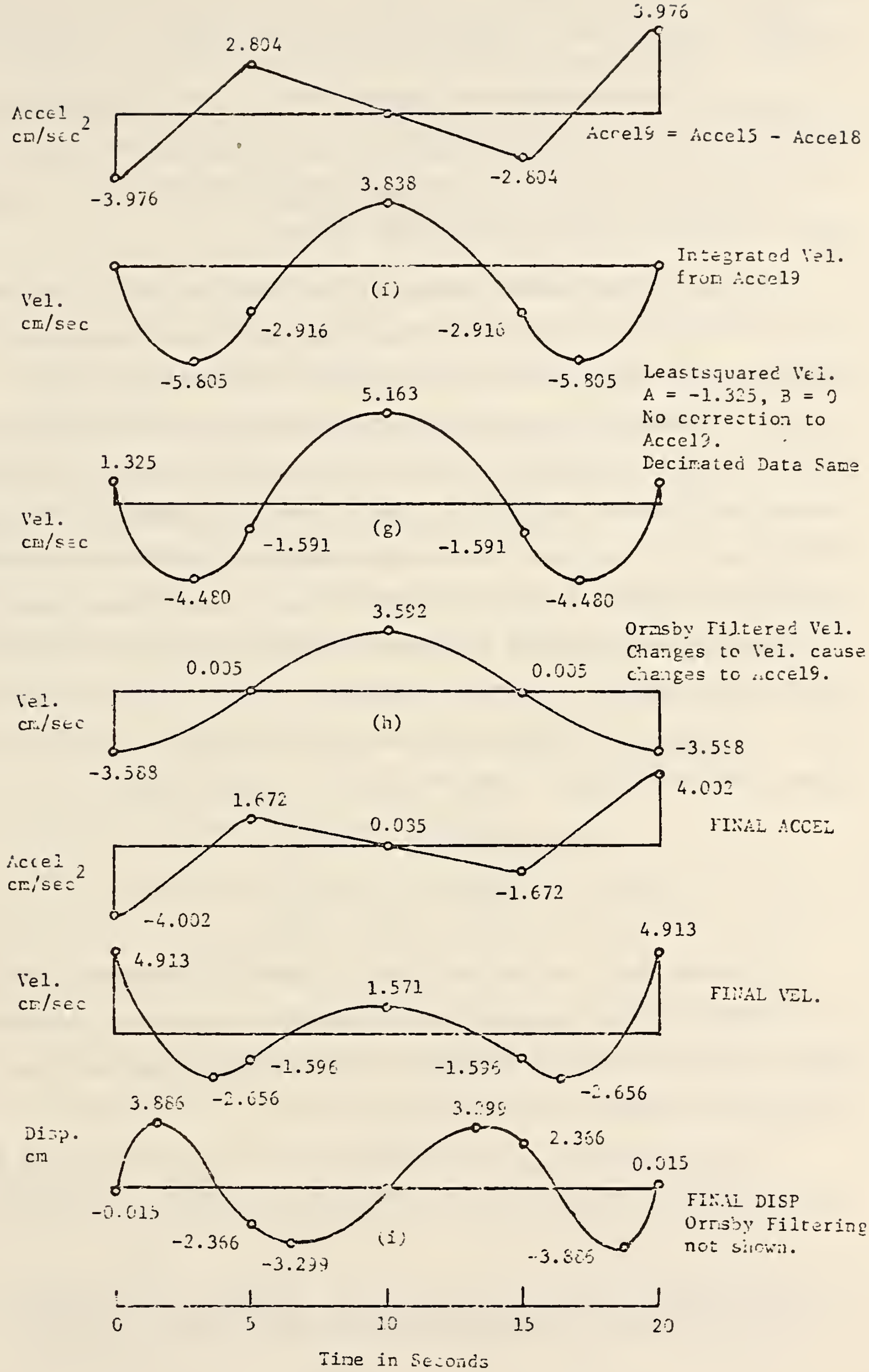


FIGURE 43 Cont.

might be slight; however, in this case the distortion completely changes the acceleration data as shown in Figure 43-c.

For completeness of information on what happens to the data in the Cal Tech program,* we shall discuss the results of the program further; although changes appear to be in order. These changes will be considered later.

The leastsquared acceleration data is next integrated to give a velocity curve of the type shown in Figure 43-c. Compare with Figure 42-b. The HOLWAY subroutine is next called to smooth out the high frequencies in the acceleration. A small change in the acceleration data occurs, as shown in Figure 43-d, particularly near the peaks and ends of the curve. In the filtering method, (the Holoway method, or equally weighted running mean filter), each data point is replaced by the average value of itself and nine points on either side of the point in question. At the ends of the curve the data is extended as an even function outside of the time interval zero to Time of Last(T) as follows:

$$a(-\tau) = a(\tau); \qquad T<\tau<0$$

$$a(T + \tau) = a(T - \tau); \quad T<\tau<2T$$

According to the Cal Tech report(16) this refinement at the ends of the acceleration curve was supposed to eliminate small errors in the velocity and displacement curves near the beginning and end of a record.

Next the Holoway filtered data is again filtered to eliminate low frequencies (long periods) using a travelling

* The program which duplicates the flowchart of reference 10 will hereafter be referred to as the Cal Tech program.

Ormsby Filter. To reduce the computer costs the Holoway data is first decimated to reduce the number of data points. Now each Ormsby filter weight is identified with (2n+1) corresponding decimated acceleration data points, where n is the number of weights in the filtering window to the right or left of the data point being replaced. Each data point is then multiplied by the corresponding Ormsby filter weight and the center acceleration point in the series is replaced by the sum of the products. Naturally the sum of the individual filter weights must be one; otherwise, the area under the filtered acceleration curve would be different than the area under the curve being filtered. Once again the acceleration curve is extended as an even function outside of its original end points. Furthermore, n filter weights cannot be greater than the number of points in the decimated curve before extension. Figure 43-e shows the filtered acceleration data, and Figure 43-f shows the resulting acceleration curve after the filtered data has been subtracted. The integrated velocity curve using the new acceleration is also shown.

When the velocity curve is integrated, small errors are reported to occur in the ground displacement.[17] Therefore, the "standard" baseline correction procedure consists of high-pass filtering the ground velocity and displacement curves, with the velocity curve extended as an even function outside the original end times before filtering. Prior to filtering the velocity curve is leastsquared, Figure 43-g. The Ormsby filtered velocity is shown in Figure 43-h. Finally, the velocity curve is integrated to give the displacement curve, which is also Ormsby filtered to remove long periods; however, in this curve the displacement curve is extended as a zero function outside of its original time span.

Figure 43-i shows the final acceleration, velocity, and displacement curves. None bear any resemblance to the calculus integrated set, Figure 42. We, therefore, now consider changes to the Cal. Tech. program to improve the situation.

The first change is to replace the leastsquaring procedure by the authors' baseline correction method (Subroutine HORIZ). In HORIZ a horizontal baseline is chosen such that the area under the acceleration curve is adjusted to zero. With this procedure we still maintain zero initial and final velocity, but do not materially disturb the shape of the original input curve. Figure 44 shows the data changes in the various steps of the revised program. To produce these changes, one merely enters a value of 1 for NEWWAY on the third control data card of PHASE2. Figure 44-a shows the input acceleration data, while Figure 44-b shows the integrated velocity. Already we see an improvement as the velocity curve is almost the same as that of Figure 42.

Figure 44-c shows the Holoway filtered acceleration data - little change. But, note the remarkable change during the Ormsby filtering, Figure 44-d. This time the Ormsby filter severely distorts the acceleration data, Figure 44-d. When this new data is integrated to obtain the velocity curve, no resemblance to the velocity curve of Figure 42 is noted. Further changes to the data by the revised program still results in a meaningless displacement curve as compared to the calculus results (Figure 42).

To ascertain if the elimination of the Ormsby filter for velocity and displacement would result in better correlation between the computer integration and the calculus integration, N2WAY was given a value of 1 in the fourth control card of PHASE2, and the data plotted through the various steps of the

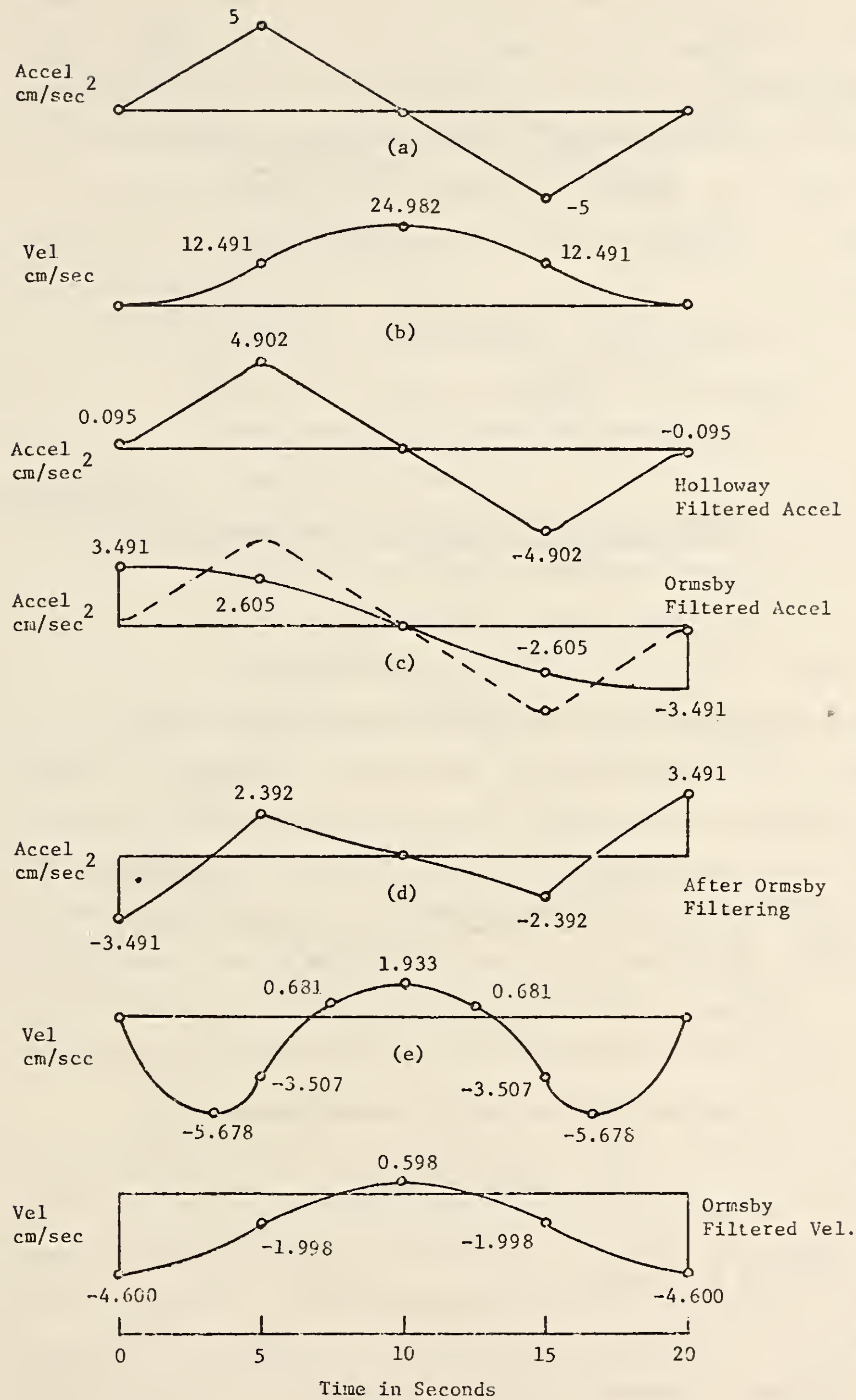


FIGURE 44 PROGRAM HISTORY - SAWTOOTH TIME-ACCELERATION CURVE - NEWWAY VERSION

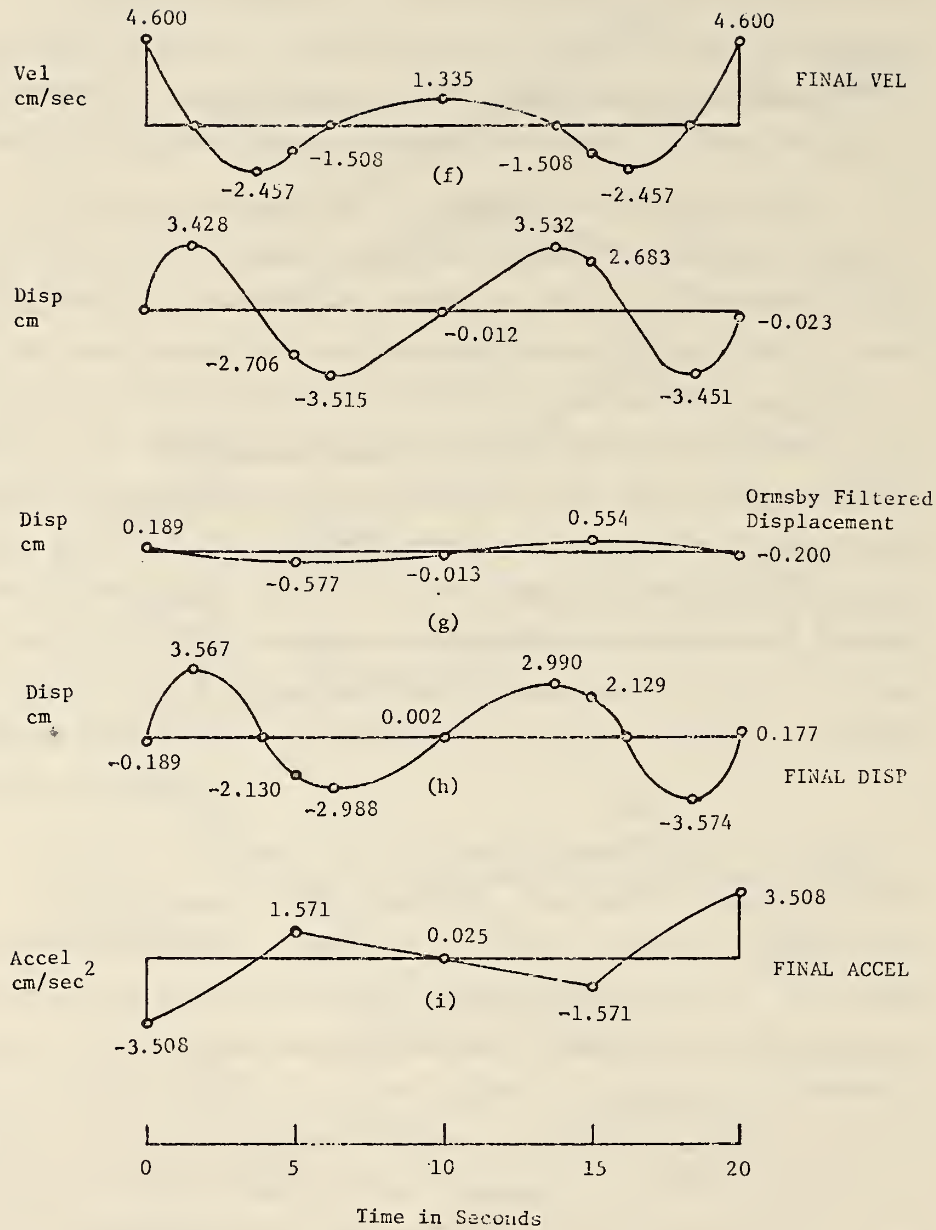


FIGURE 44 Cont.

numerical integration, Figure 45. Sure enough, the displacement data is now beginning to look more like the corresponding data of Figure 42;.but of a much lower amplitude. The problem lies with the Ormsby filter on the acceleration.

We have now reduced the computer program to a much earlier version by the authors of the Cal. Tech. program [9] except for two changes:

A. Reference 9 considered the acceleration data to be zero outside of the original time domain for both the Holoway and Ormsby filtering, and

B. Reference 9 used the leastsquaring procedure for adjusting the acceleration baseline instead of the procedure now used by subroutine HORIZ.

It is interesting to note that if the Holoway and Ormsby filters were eliminated for the acceleration curve, the Cal. Tech. version in reference 9 would become similar to the Boyce computer program.[7] Minor differences exist in the manner of leastsquaring: Boyce (New Zealand) uses a parabolic baseline correction on velocity, while Trifunac (Cal. Tech.) uses a straight line correction on acceleration. Details of leastsquaring for both programs are described in Appendix A.

What would happen if the Holoway and Ormsby Filters were eliminated on the acceleration curve?. Would we get the identical curves to Figure 42? To accomplish this a new version N3WAY was given a value of 1 on the third input control card of PHASE2. The results are shown in Figure 46. They are identical to Figure 42. We have now returned to a computer program similar to Housner's original version, except for the manner of leastsquaring of the acceleration data.[18]

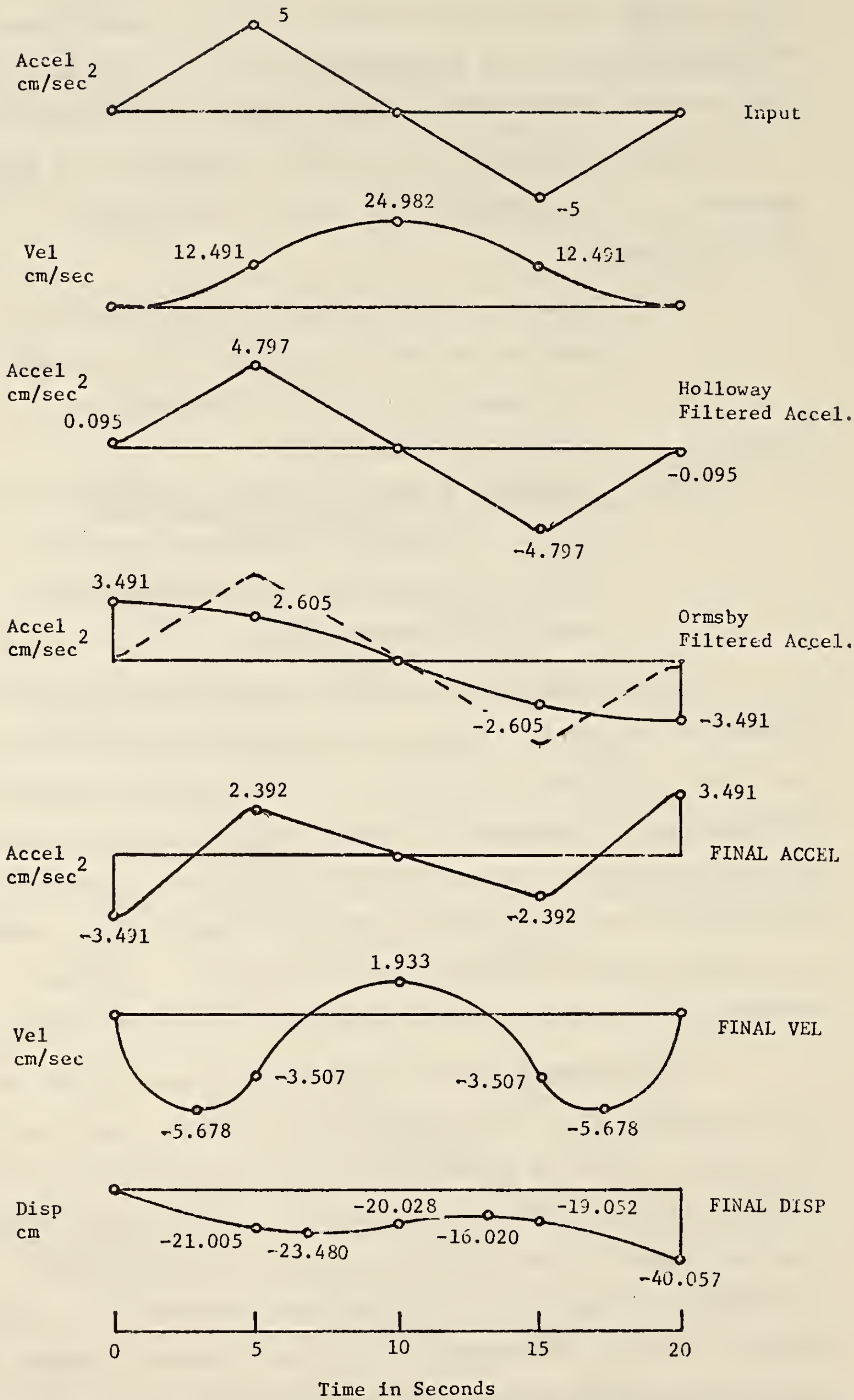


FIGURE 45 PROGRAM HISTORY - SAWTOOTH TIME-ACCELERATION CURVE - N2WAY VERSION

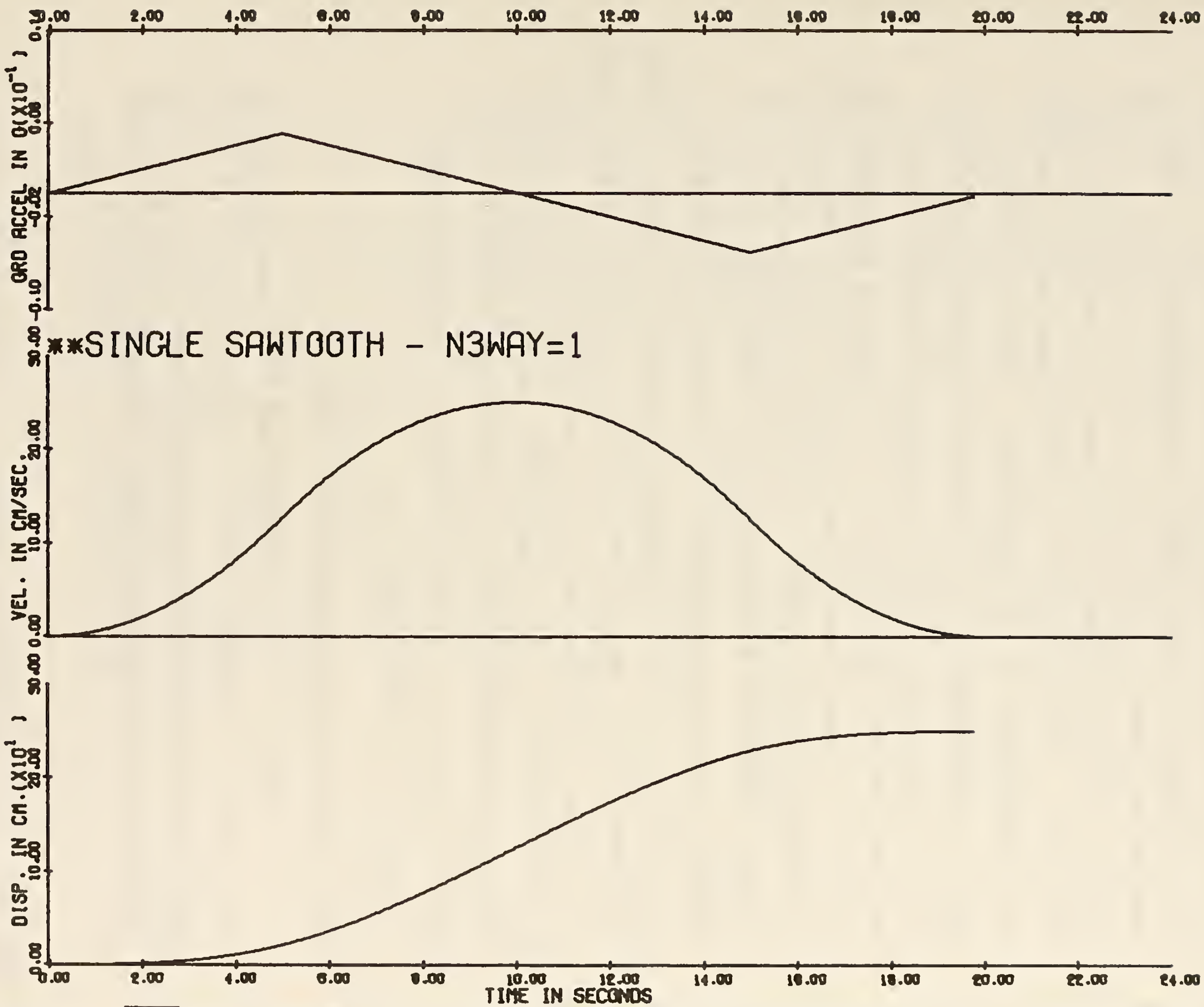


FIGURE 46 PHASE2 RESULTS FOR SINGLE SAWTOOTH TIME-ACCELERATION DATA -- N3WAY VERSION

4. Cosine Curve

In preparation for an experiment to be described in the next chapter approximating a cosine variation, a seven-cycle cosine curve was integrated using N3WAY = 1. The results are shown in Figure 47. A careful scrutiny of the computer printout showed the problem to occur with the last data point. The curve goes to 5.83 seconds (approximately the time duration of the experiment). When the original data is interpolated to 0.01 seconds everything goes well, but when every other point is discarded (data now at 0.02 seconds) the 5.83 point is lost. The baseline adjustment is in error by a small amount, but enough to give a larger error in the displacement record. Accordingly, the baseline adjustment was made while the data was interpolated at 0.01 seconds; then every other point was discarded (N5WAY = 1).* The result is excellent as shown in Figure 48. Figures 47 and 48 illustrate the importance of careful attention to end points in an array. Only one point made the difference; put Figures 47 and 48 together over a light table and no apparent difference is discernable in the acceleration and velocity curves. Figure 49 shows the results of using the basic Cal. Tech. program for the same cosine curve.

5. Justification of Leastsquaring and Filtering

To deal with pure mathematical functions ignores the true nature of earthquake records; they don't occur as precise mathematical functions, they are more random oriented. A true horizontal acceleration such as in Figure 40 will not record as a horizontal acceleration. In the first place, there is

* N4WAY = 1 is a variation not germane to this discussion and is described in Appendix C, p. C-10.

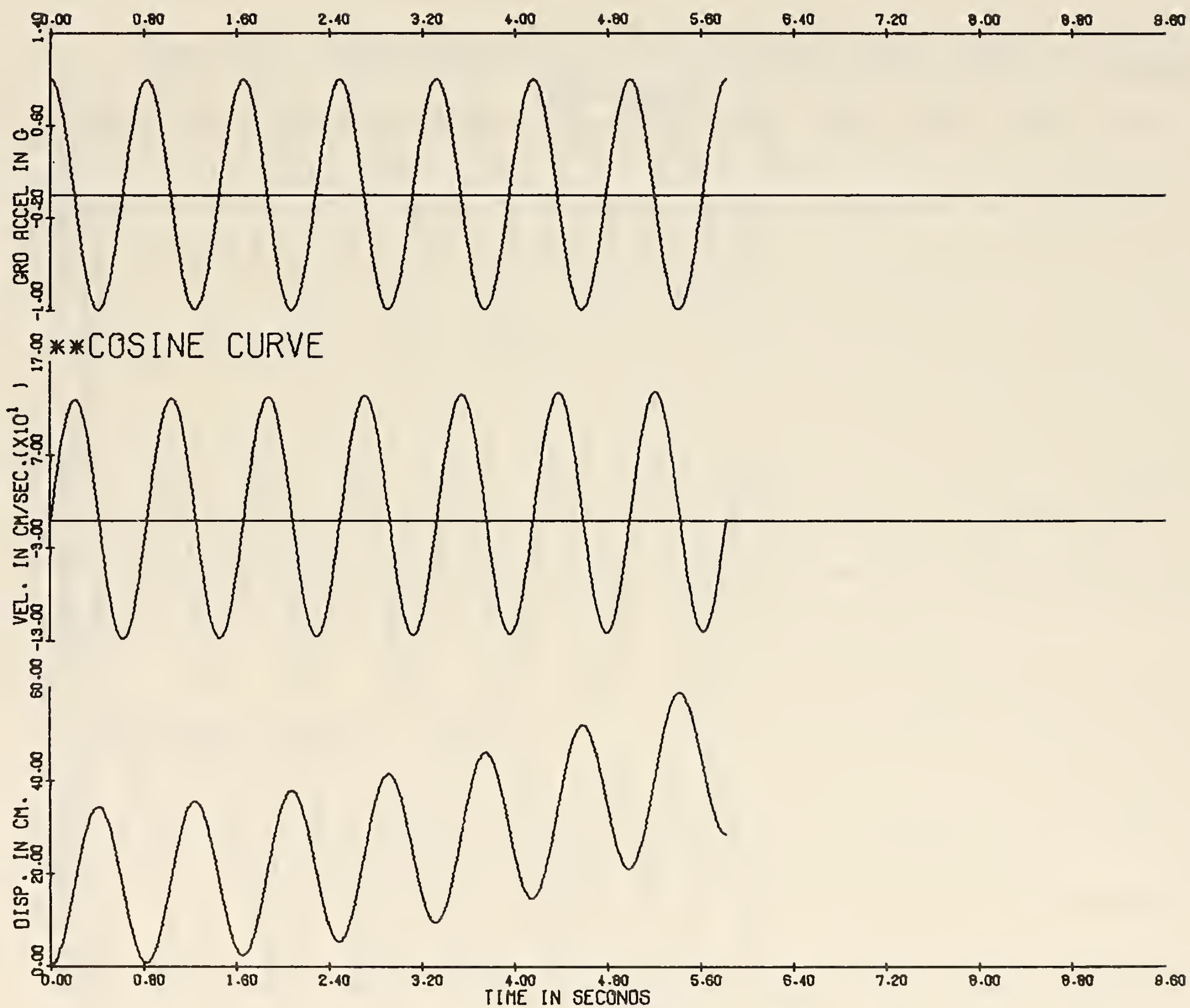


FIGURE 47 PHASE2 RESULTS FOR 7 CYCLE COSINE CURVE - N3WAY VERSION

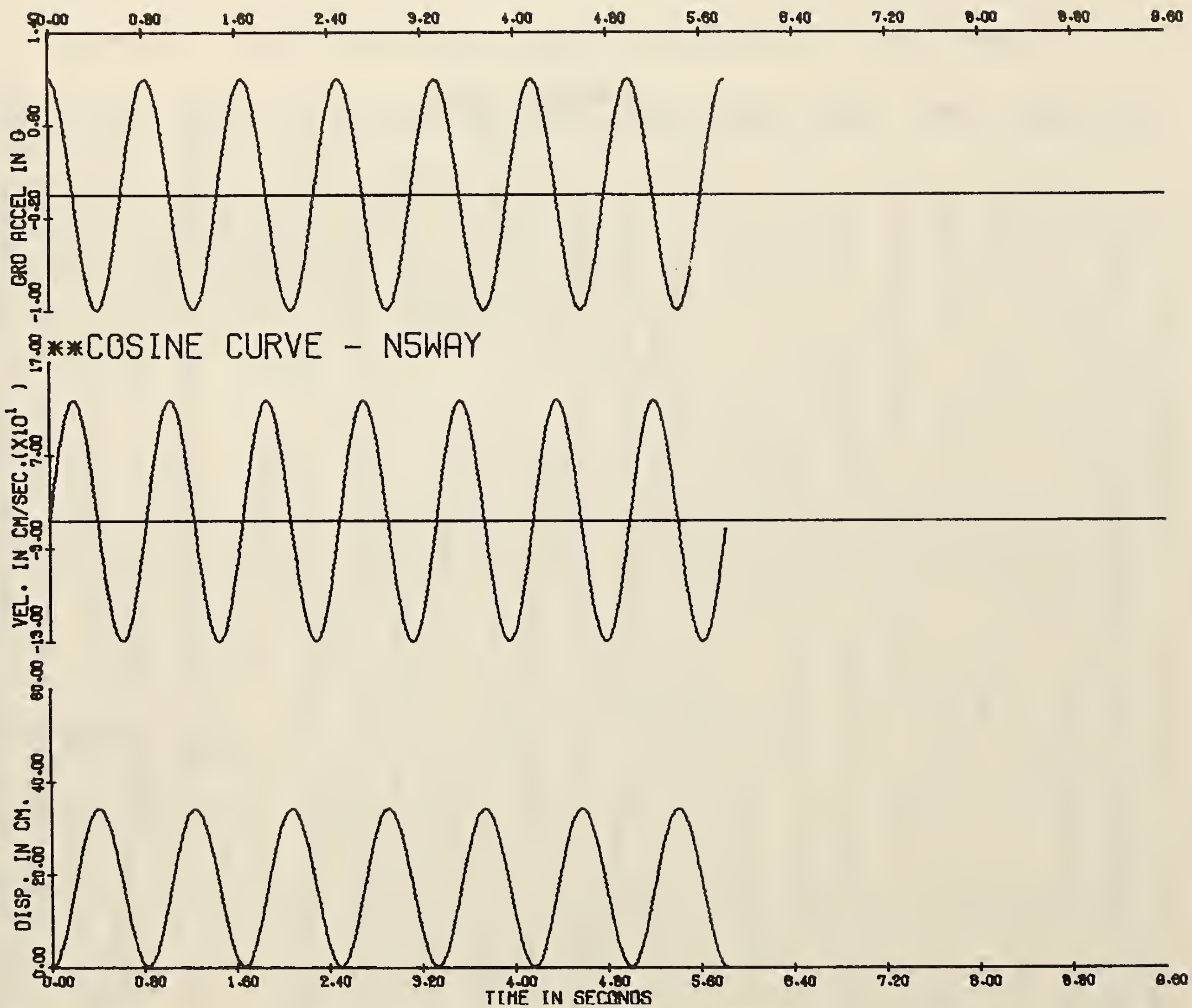


FIGURE 48 PHASE2 RESULTS FOR 7 CYCLE COSINE CURVE - N5WAY = 1 VERSION

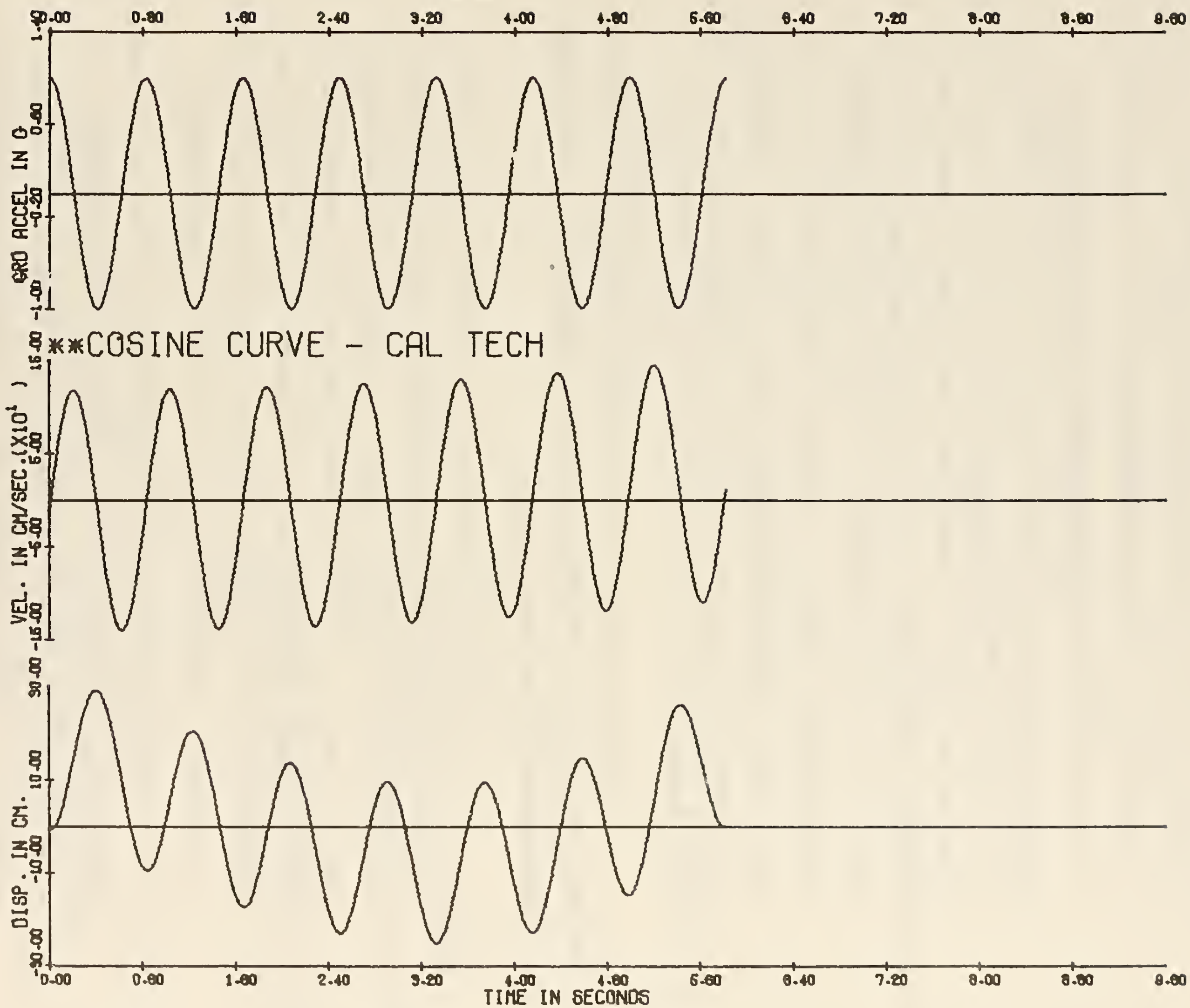


FIGURE 49 PHASE2 RESULTS FOR 7 CYCLE COSINE CURVE - CAL. TECH. VERSION

start-up time to be considered as an accelerometer system is turned on. The horizontal straight line could be recorded as a sloping curve (if not altogether straight) as the instrument is warming up. The leastsquaring method now becomes mandatory to correct for the instrumentation error by adjusting the sloping recorded acceleration curve to one more resembling the true time-acceleration history. In the second place, the time delay at the start of recording (see Figure 1) means that we have lost the starting zero acceleration value. With the end point of the record possibly not defined because of drift in the recording we have no alternative but to turn to an adjustment method like leastsquaring to try to locate a baseline that makes the beginning and end velocities zero; end conditions we do know.

Cyclic variations in the recording due to instrument drift or later in PHASE1 due to the digitization processing also makes filtering mandatory.

In effect we can say that if the record is perfect, the leastsquaring and filtering schemes will distort the record because they are trying to eliminate something that wasn't there; but if the record does contain errors, then these schemes hopefully are of a nature as to mitigate the continuation of the errors when the time-displacement history is obtained. We are now back to the Cal. Tech. program.

Previously we have discussed how errors due to least-squaring and filtering are introduced in short records. It will be profitable to further examine their production in long records, say thirty seconds; and if possible to determine a change in the program to eliminate them. But first we should also examine the nature of the integration process.

6. Nature of Integration

Consider an acceleration sine curve of the form:

$$A = \sin c\pi t$$

By integration the velocity becomes

$$v = \frac{1}{c} \sin c\pi t + C_1$$

If the velocity is zero at time zero, $C_1 = 0$. The displacement becomes:

$$d = \frac{1}{c^2\pi^2} \sin c\pi t + C_2$$

If the displacement is zero at time zero, $C_2 = 0$. Notice the following conditions:

If $c^2\pi^2$	< 1.0	The amplitude of displacement is greater than the amplitude of acceleration.
	$= 1.0$	The amplitude of displacement equals the amplitude of acceleration.
	> 1.0	The amplitude of displacement is less than the amplitude of acceleration.

To put a scale on thse effects let us consider the frequency range between .1 Hz and 12 Hz.* If we take an average frequency, say 5 Hz at a peak acceleration of 1 G, then the peak displacement will be $\frac{1}{10^2\pi^2}$ 980.665 = .994 cm or about $\frac{1}{1000}$ the magnitude of the peak acceleration. Now let the 5 Hz extend over a thirty second record, where a leastsquaring

* Reference (19) states that "a strong motion accelerometer should record accurately over a period range of 0.1 to at least three or four seconds and maybe to ten seconds."

acceleration end changes of say 1 cm/sec^2 occurs. The peak displacement due to that long period change will now be

$$\frac{1}{(\frac{1}{60})^2 \pi^2} (1) = 364.757 \text{ cm.}$$

The long period now completely dominates the time-displacement curve while the 5 Hz frequency is reduced to a low level noise curve superimposed on the lower frequency. We are now at the situation where the computer program introduces long period errors into a perfect record and integrates these errors as the final displacements - unless, of course, we high-pass filter the displacement record above the error frequency level.

7. .625 Hz Sine Curve

Figure 50 shows the integrated acceleration, velocity, and displacement curves by the Cal. Tech. program for an input sine acceleration curve of period 1.6 seconds and peak amplitude 4.836 cm/sec^2. Essentially the input data is a .625 Hz sine curve extending over a time domain of 28.8 seconds. Thus there are 18 full cycles to the input data. To follow the changes in the data we shall now show printer plots at the various stages of the program. Printer plots have a background grid, the computer paper, which makes it easier to read small changes in the data than the Calcomp Plotter plots, and is much cheaper to produce. Figure 51-a shows the plot of about 1/2 the input data created at 0.01 seconds. The data has an even number of spaces per 1/4 cycle, hence there is no problem with peaks remaining unchanged when the data is reduced to 0.02 second intervals. Figure 51-b shows the full record at 0.01 second intervals. After leastsquaring

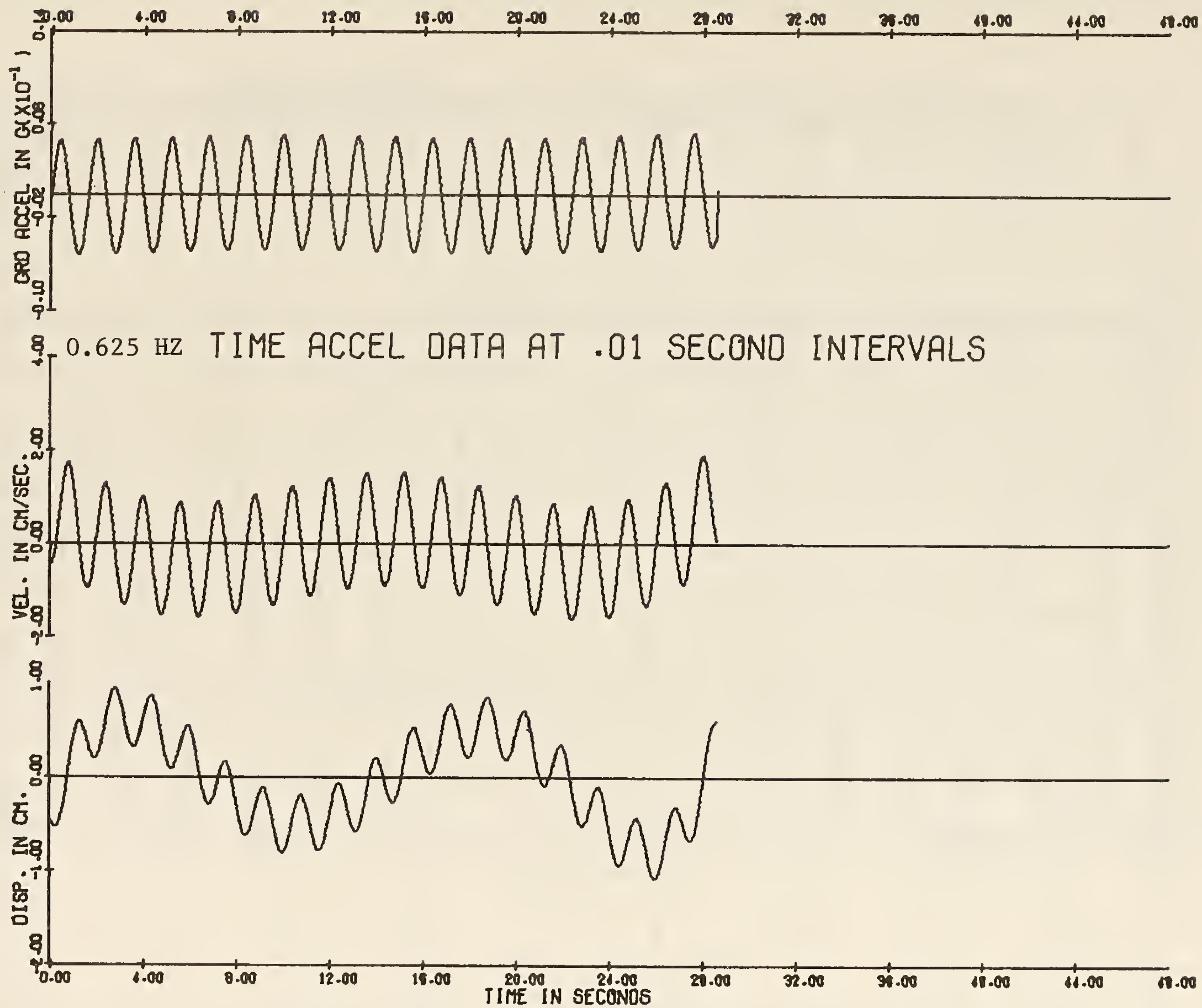


FIGURE 50 PHASE2 RESULTS FOR .625 HZ SINE CURVE - CAL. TECH. VERSION

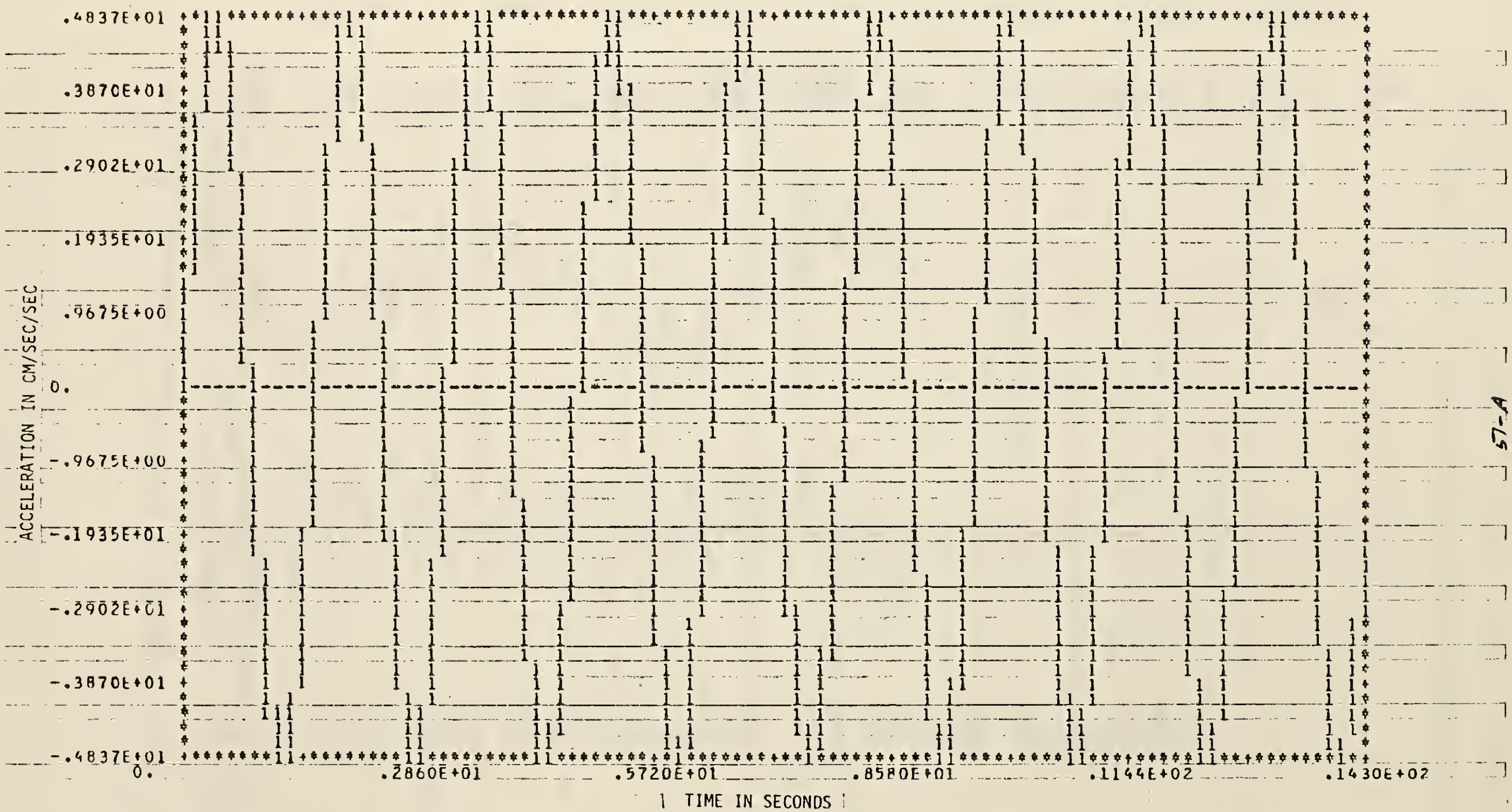


FIGURE 51-a PRINTER PLOT OF .625 HZ SINE CURVE AT INPUT DATA, 0.01 SECOND INTERVALS

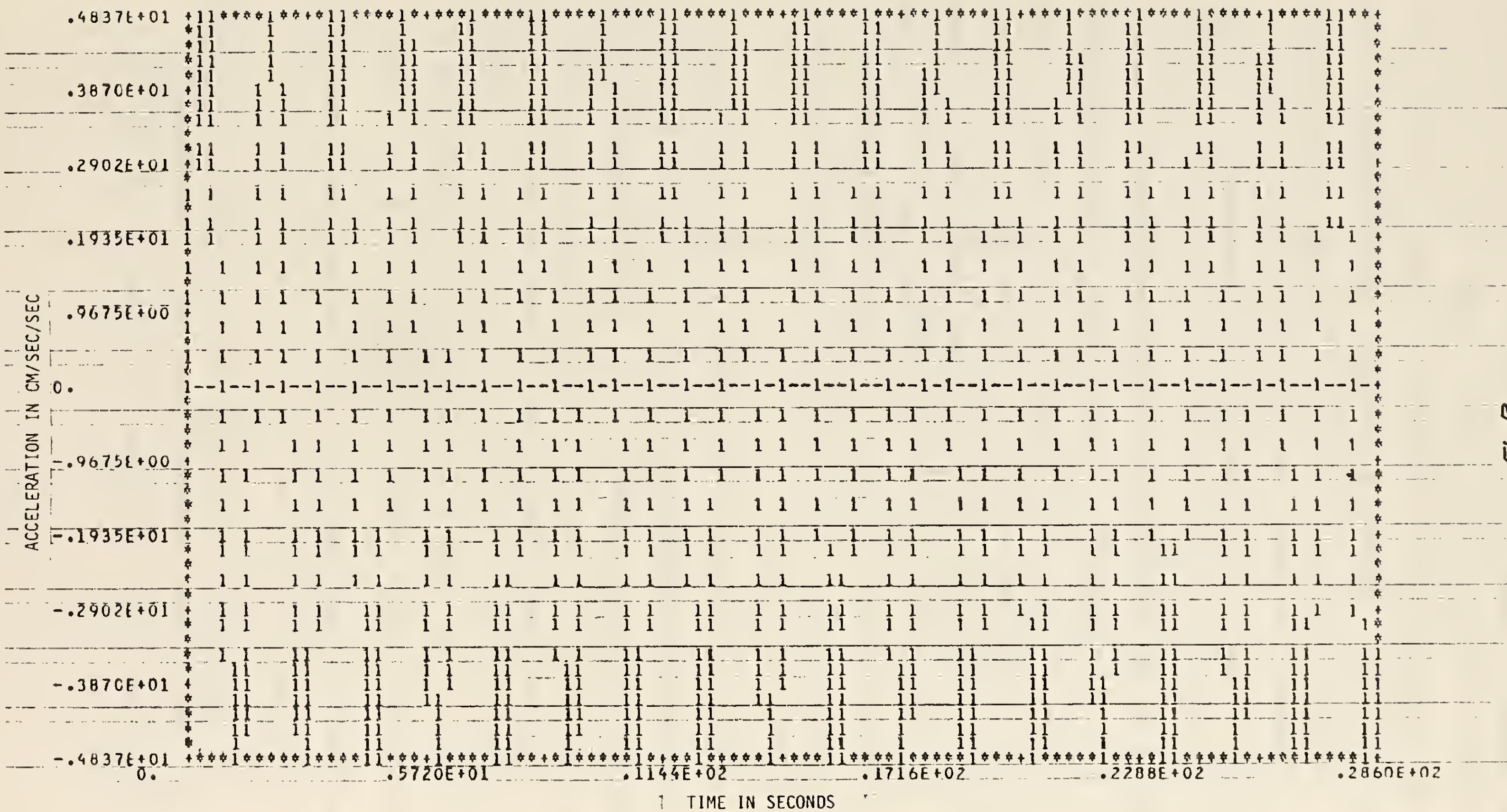


FIGURE 51-b PRINTER PLOT OF .625 HZ SINE CURVE DECIMATED TO 0.02 SECOND INTERVALS

the new baseline equation is A = .2565 - 0.0178t, indicating an immediate change in the record, albeit small. Figure 51-c shows the leastsquared acceleration. Figure 51-d shows the acceleration after Holoway filtering. The changes are evident. The acceleration curve tends to slope upward to the right. Figure 51-e shows the acceleration after Ormsby filtering. A small cyclic variation is introduced. Figure 51-f shows the final acceleration after correction because of changes in the velocity due to leastsquaring the velocity. Figure 51-g illustrates the leastsquared velocity prior to filtering, while Figure 51-h shows the final velocity after filtering. These large cyclic variations in velocity lead to tremendous changes in the integrated displacement, Figure 51-i. Further changes in the displacement due to filtering at .067 Hz do little to correct the final displacement as shown in Figure 51-j. Notice that the long period curve is approximately at .07 Hz.

Figure 52 shows an improvement in the situation as the Ormsby filter frequency is raised to 0.1 Hz. (This is accomplished by letting N4WAY = 7.) The displacement pattern is starting to straighten out. We are starting to filter above the error band frequencies. Figure 53 shows excellent results as the filter frequency is raised to 0.3 Hz (N4WAY = 8). We expect the final displacement by calculus to be a sine curve with a peak value of $\frac{4.836}{1.25^2 \pi^2} = .313$ cm. The printout shows a peak value of 0.331 cm. except at the last cycle which has a maximum negative peak of 0.372 cm.

If our theory is correct, and the filters are working properly then the displacement record should practically vanish when the filter frequency is raised above .625 Hz.

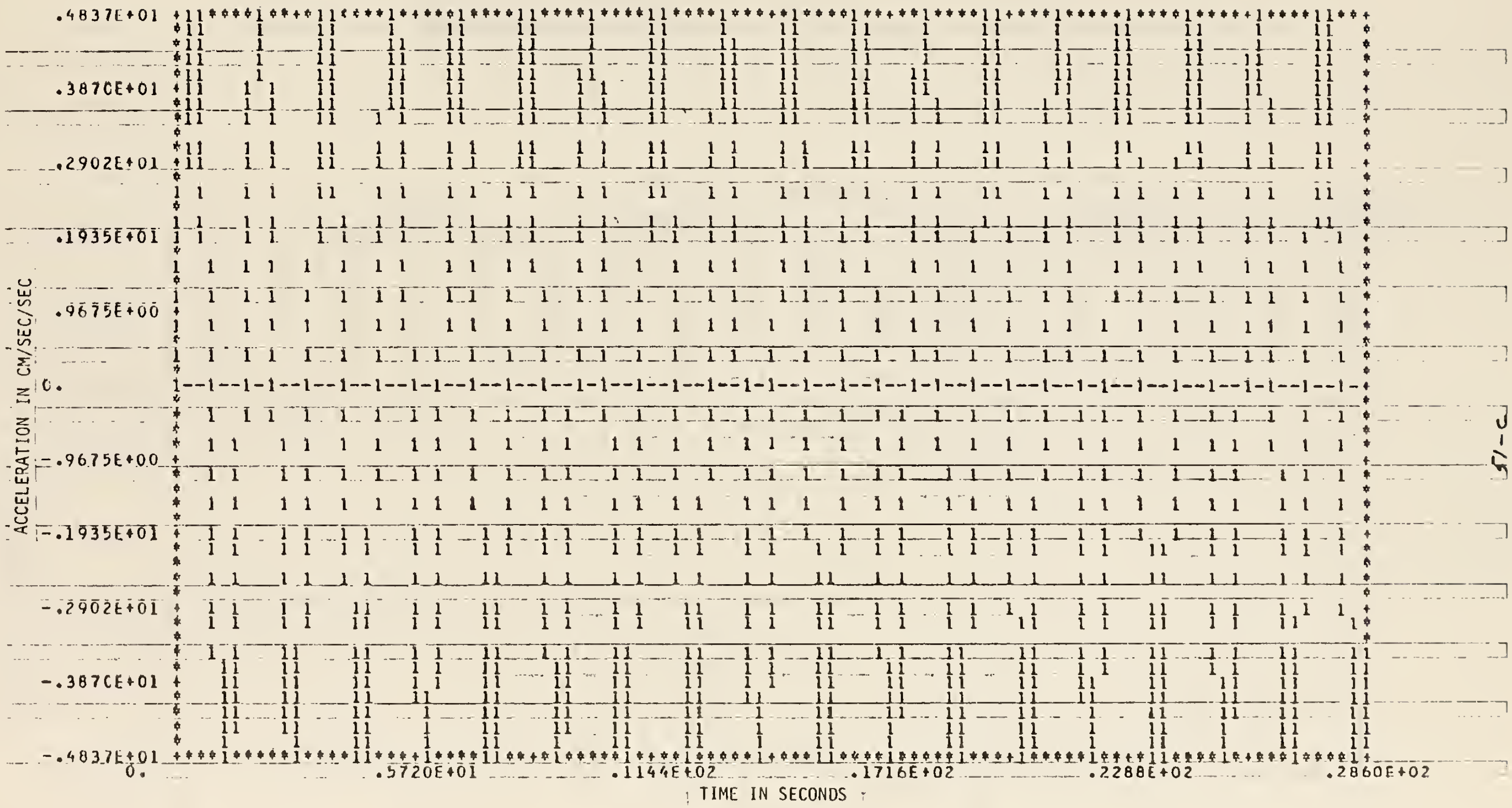


FIGURE 51-c PRINTER PLOT OF .625 HZ SINE CURVE - ACCELERATION AFTER LEASTSQUARING

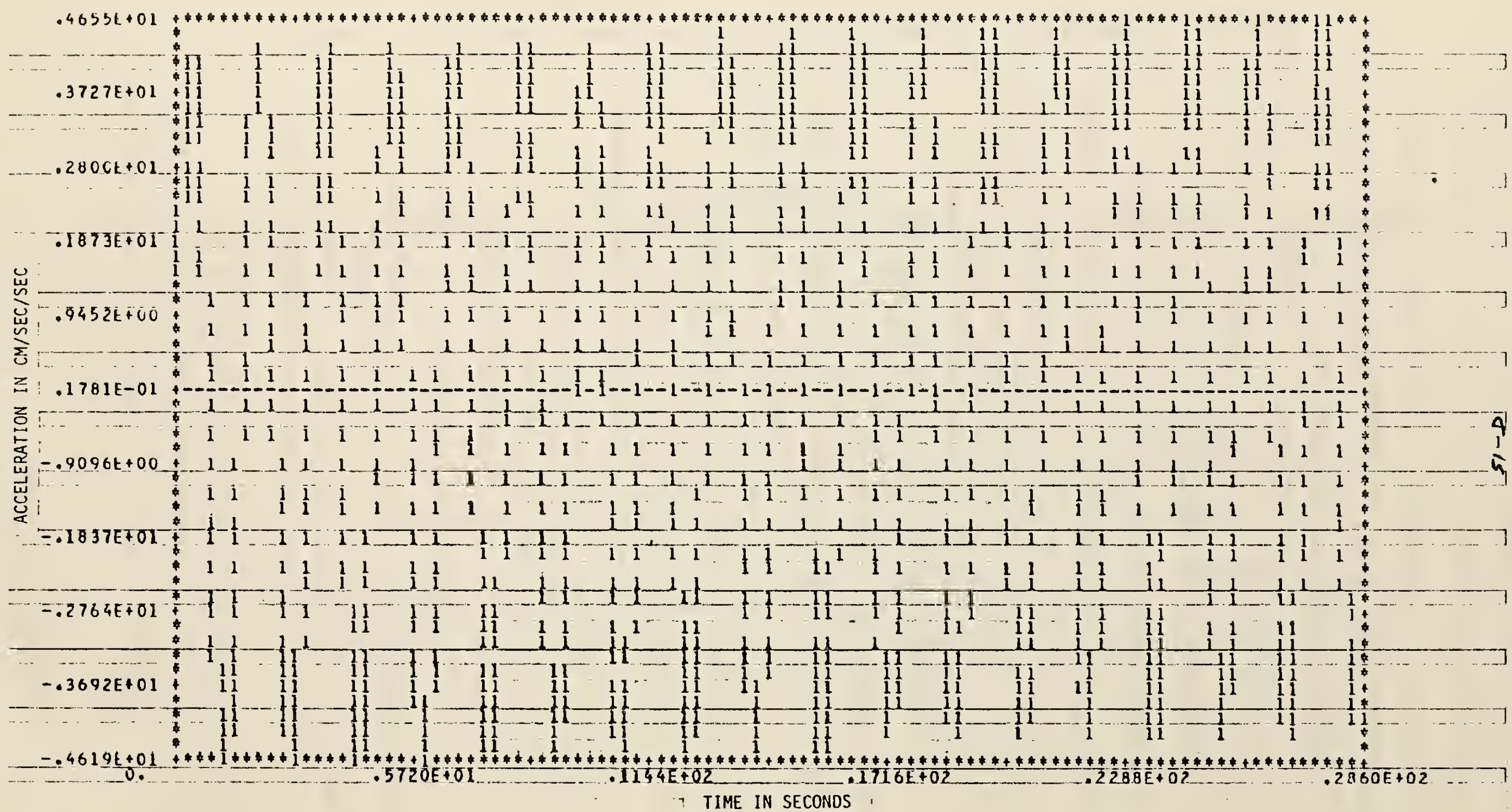


FIGURE 51-d PRINTER PLOT OF .625 HZ SINE CURVE - ACCELERATION AFTER HOLOWAY FILTERING

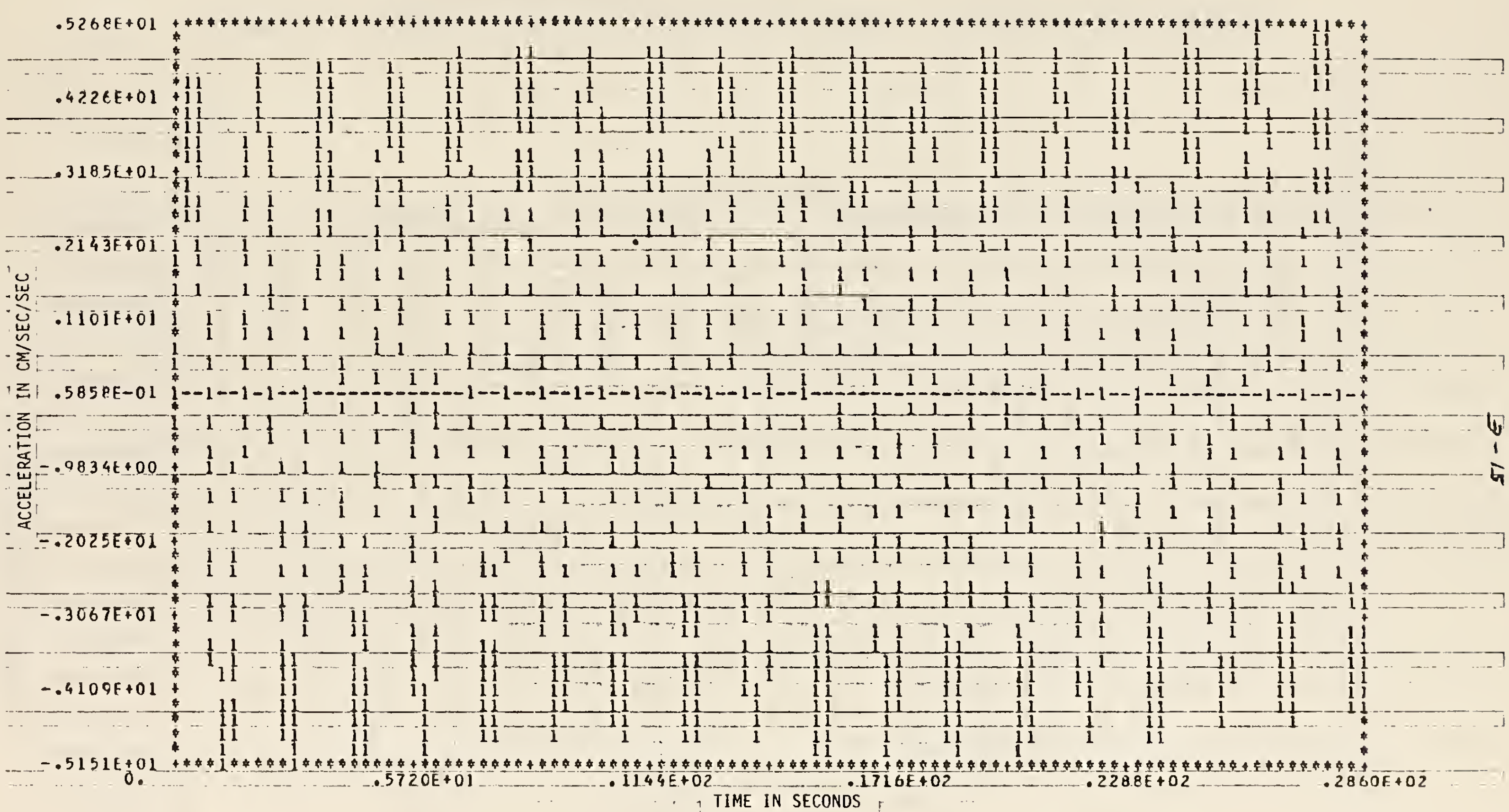


FIGURE 51-e PRINTER PLOT OF .625 HZ SINE CURVE - ACCELERATION AFTER ORMSBY FILTERING

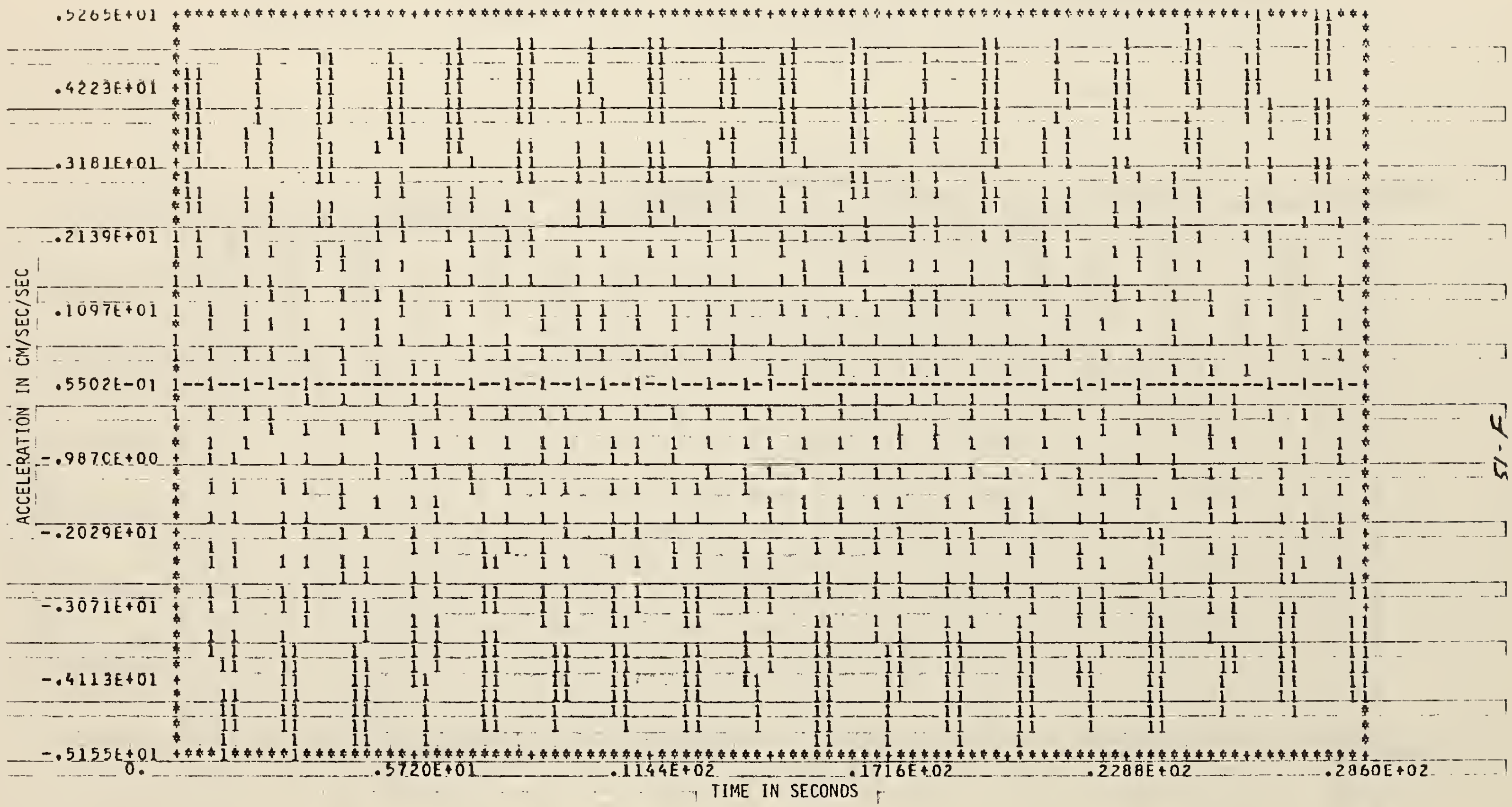


FIGURE 51-f PRINTER PLOT OF .625 HZ SINE CURVE - ACCELERATION CORRECTED AFTER LEASTSQUARING VELOCITY

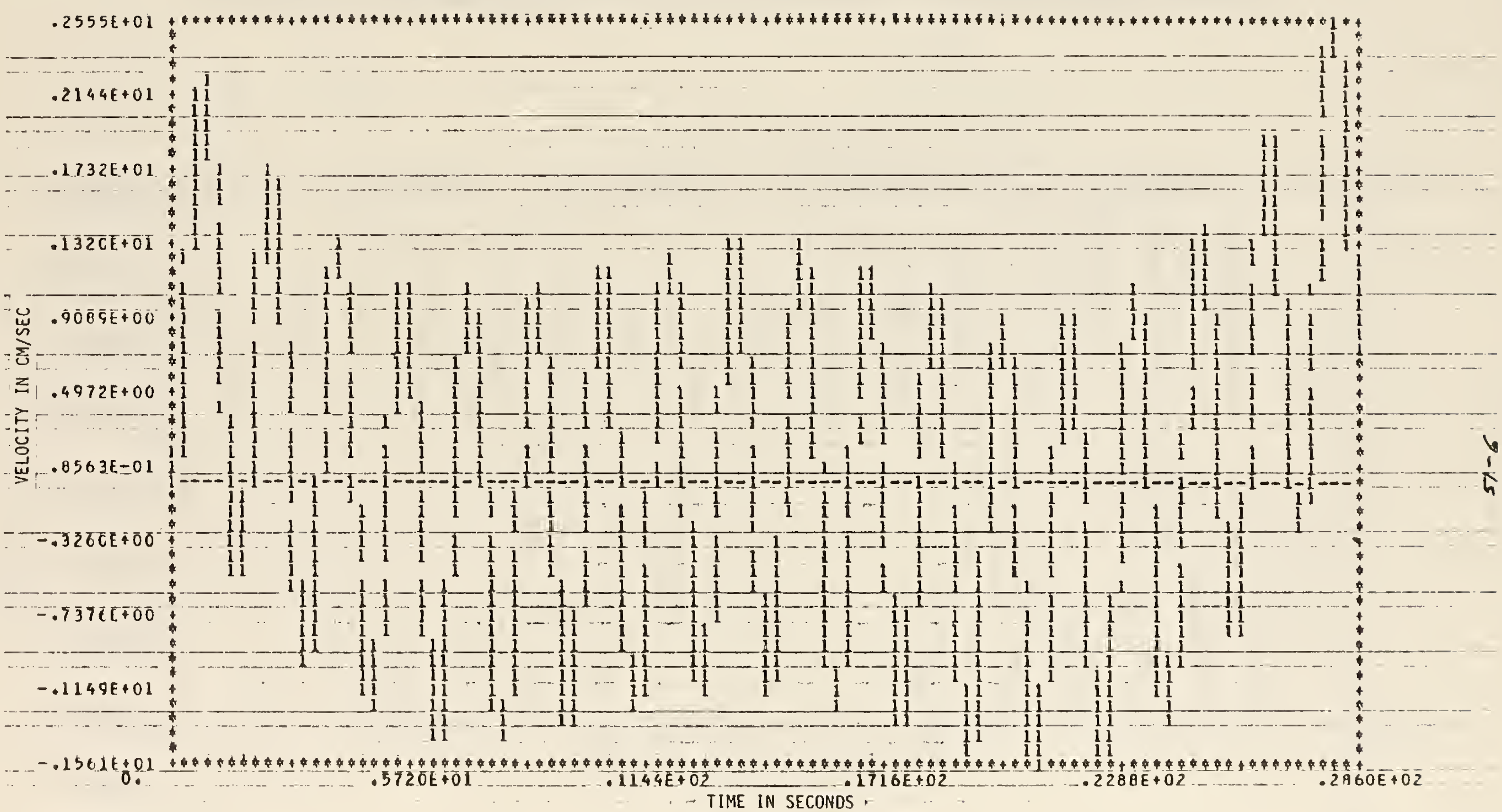


FIGURE 51-g PRINTER PLOT OF .625 HZ SINE CURVE - LEASTSQUARED VELOCITY PRIOR TO FILTERING

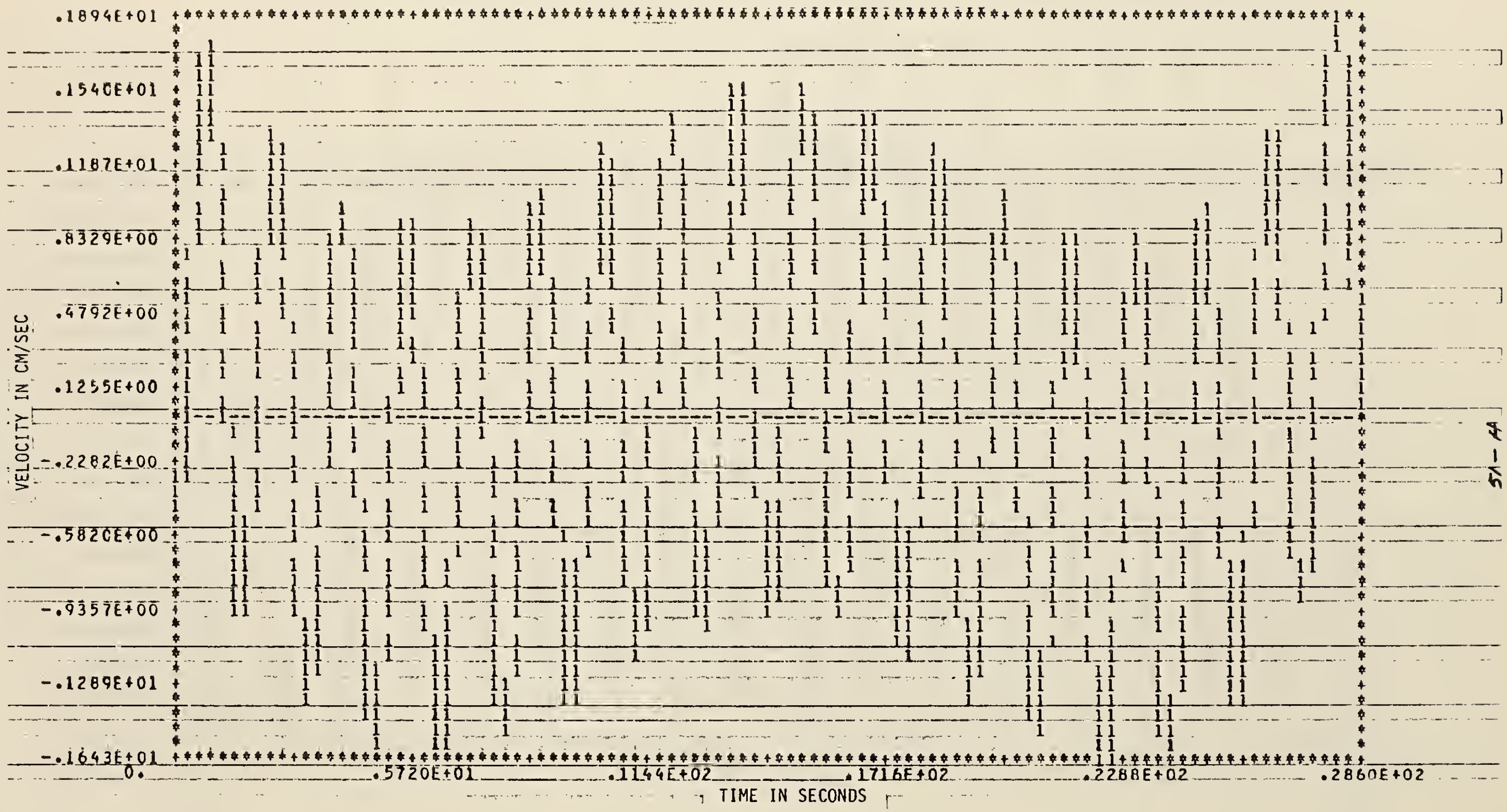


FIGURE 51-h PRINTER PLOT OF .625 HZ SINE CURVE - FINAL VELOCITY AFTER FILTERING

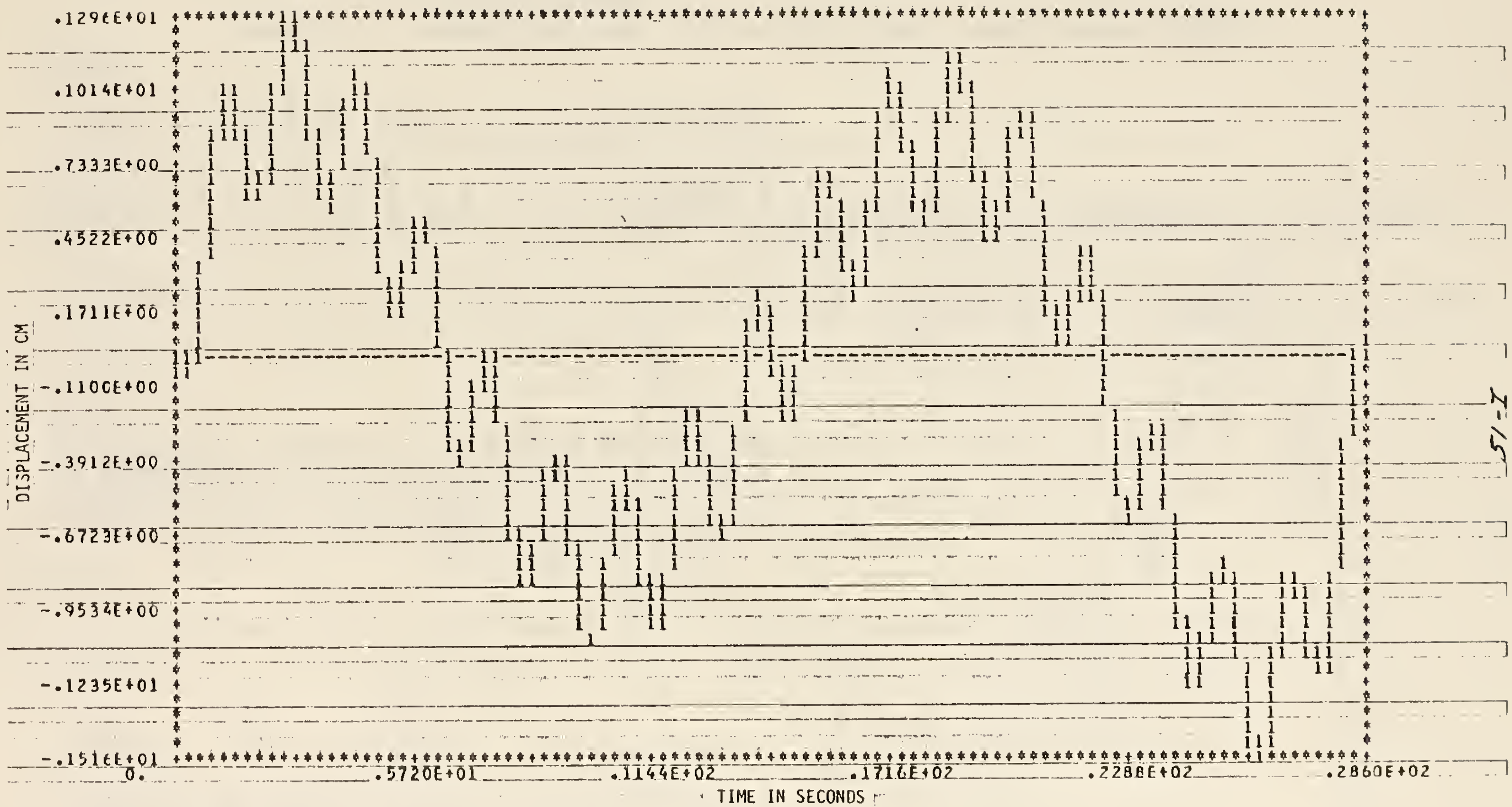


FIGURE 51-i PRINTER PLOT OF .625 HZ SINE CURVE - INTEGRATED DISPLACEMENT PRIOR TO FILTERING

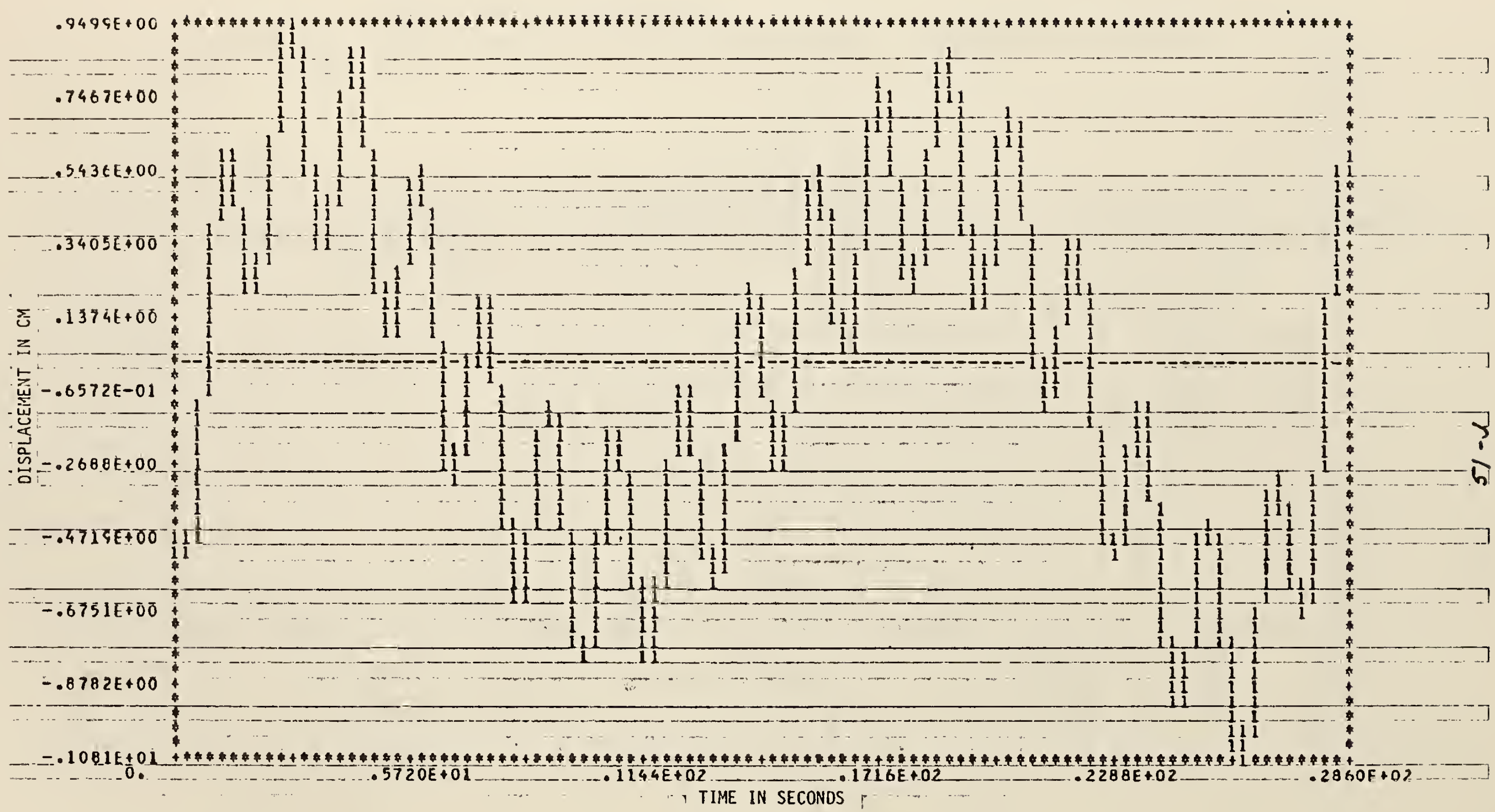


FIGURE 51-j PRINTER PLOT OF .625 HZ SINE CURVE - FINAL DISPLACEMENT AFTER FILTERING

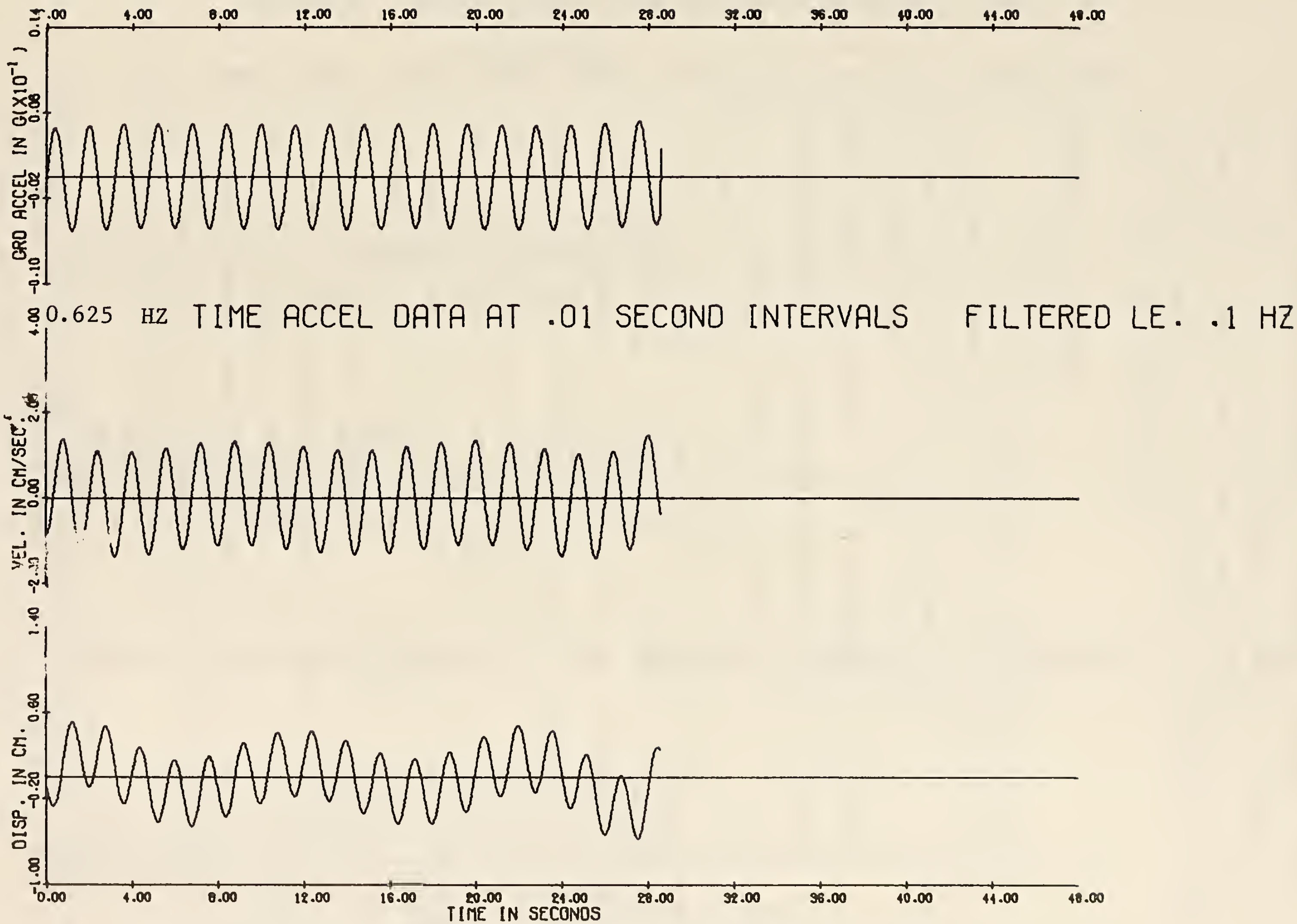


FIGURE 52 PHASE2 RESULTS FOR .625 HZ SINE CURVE - N4WAY = 7

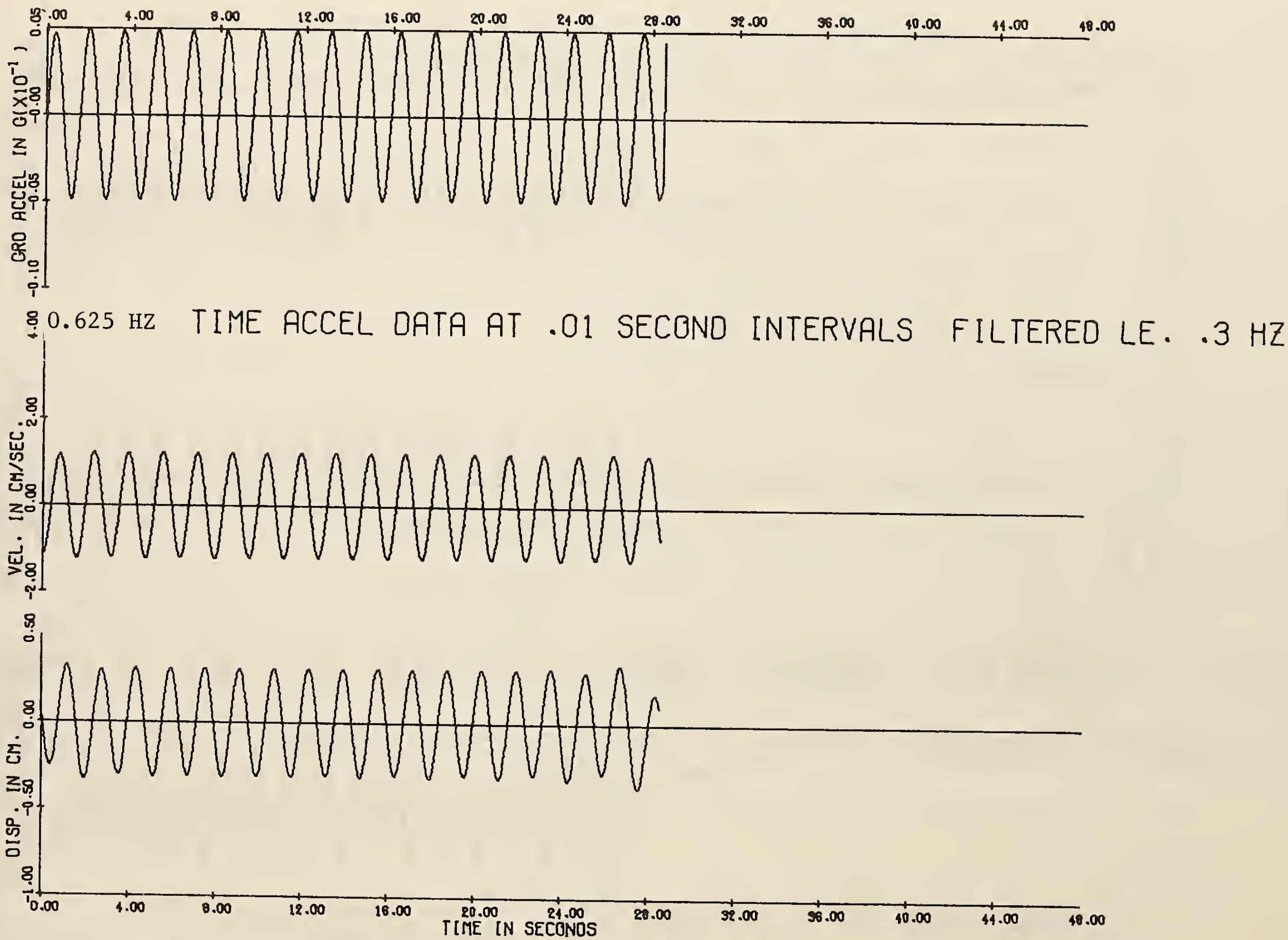


FIGURE 53 PHASE2 RESULTS FOR .625 HZ SINE CURVE - N4WAY = 8

Figure 54 shows the results of changing the Ormsby filter to 1 Hz (N4WAY = 10). The displacement record is wiped out as predicted. Only an end noise remains skewed to the right of the record because of the antisymmetric arrangement of input data about the time centerline.

8. Symmetrical Sawtooth

To see what happens when a long period record is evaluated, a symmetrical sawtooth acceleration record was processed through the Cal. Tech. program, Figure 55. As expected the displacement record shows a symmetrical long period. This time there is no change in the acceleration data due to leastsquaring. The correction baseline has the formula A = 0 + 0t. Figure 56 shows the changes by means of printer plots. Figure 56-a shows the acceleration data prior to Holoway filtering, and Figure 56-b shows the data after Holoway filtering. Very little change has occurred. Figure 56-c shows the Ormsby filtered data and Figure 56-d the acceleration after subtraction of the filter; a tremendous change in the shape of the acceleration curve. This shows why it would have been impractical to try to change Figure 43 by the simple expedient of changing the filter frequency. The result would be to completely wipe out the entire record. Figure 56-e shows the integrated velocity and Figure 56-f the final velocity. Figure 56-g shows the integrated displacement and Figure 56-h the filtered displacement with a .067 Hz filter. This filter frequency hardly makes any difference in the long period displacement curve. Contrast these previous results with the calculus integration, Figure 57.

Now lets see what happens when the Ormsby filter frequency is changed to 1 Hz, Figure 58. As expected the velocity and displacement records are completely wiped out,

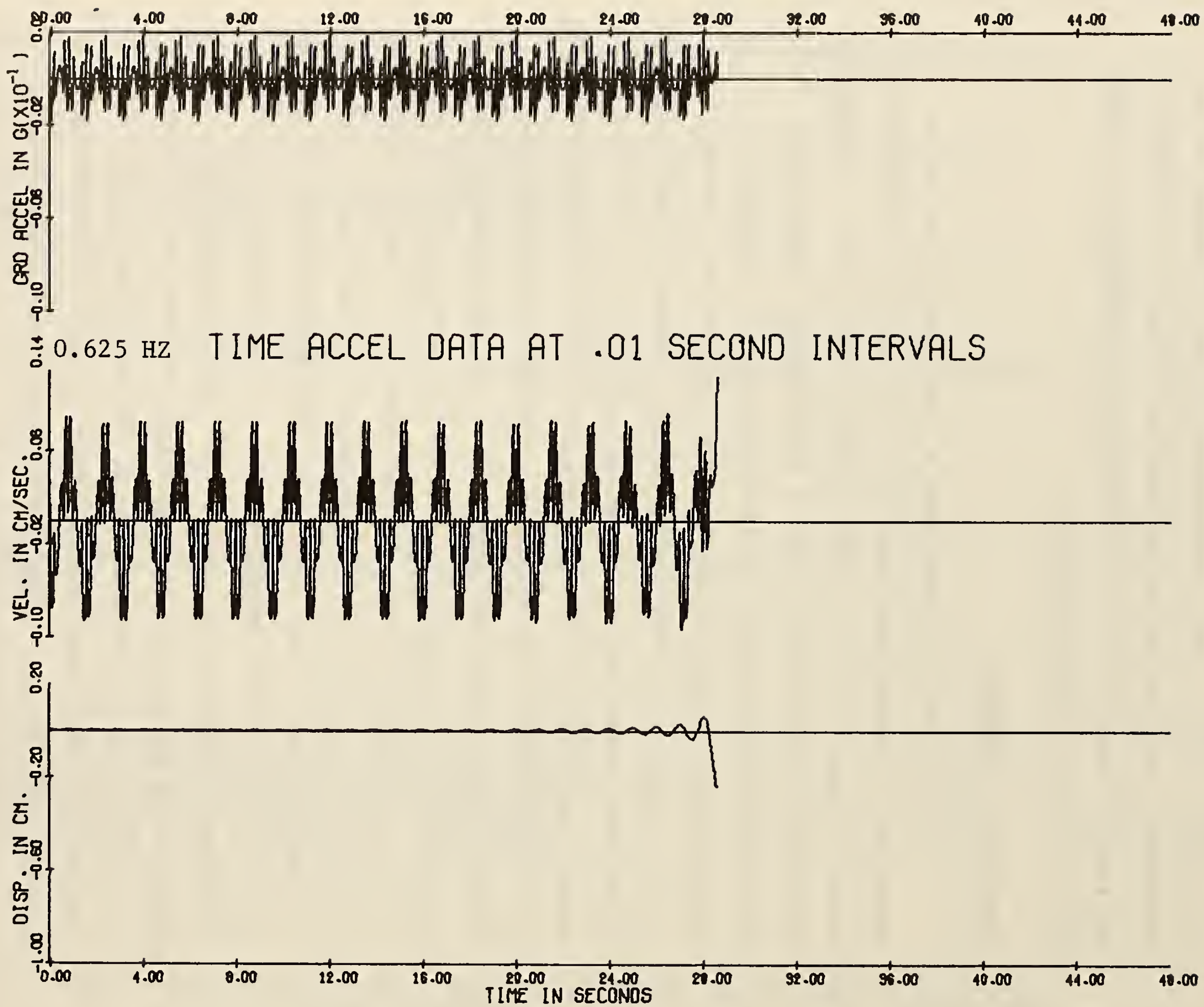


FIGURE 54 PHASE2 RESULTS FOR .625 HZ SINE CURVE ~ N4WAY = 10

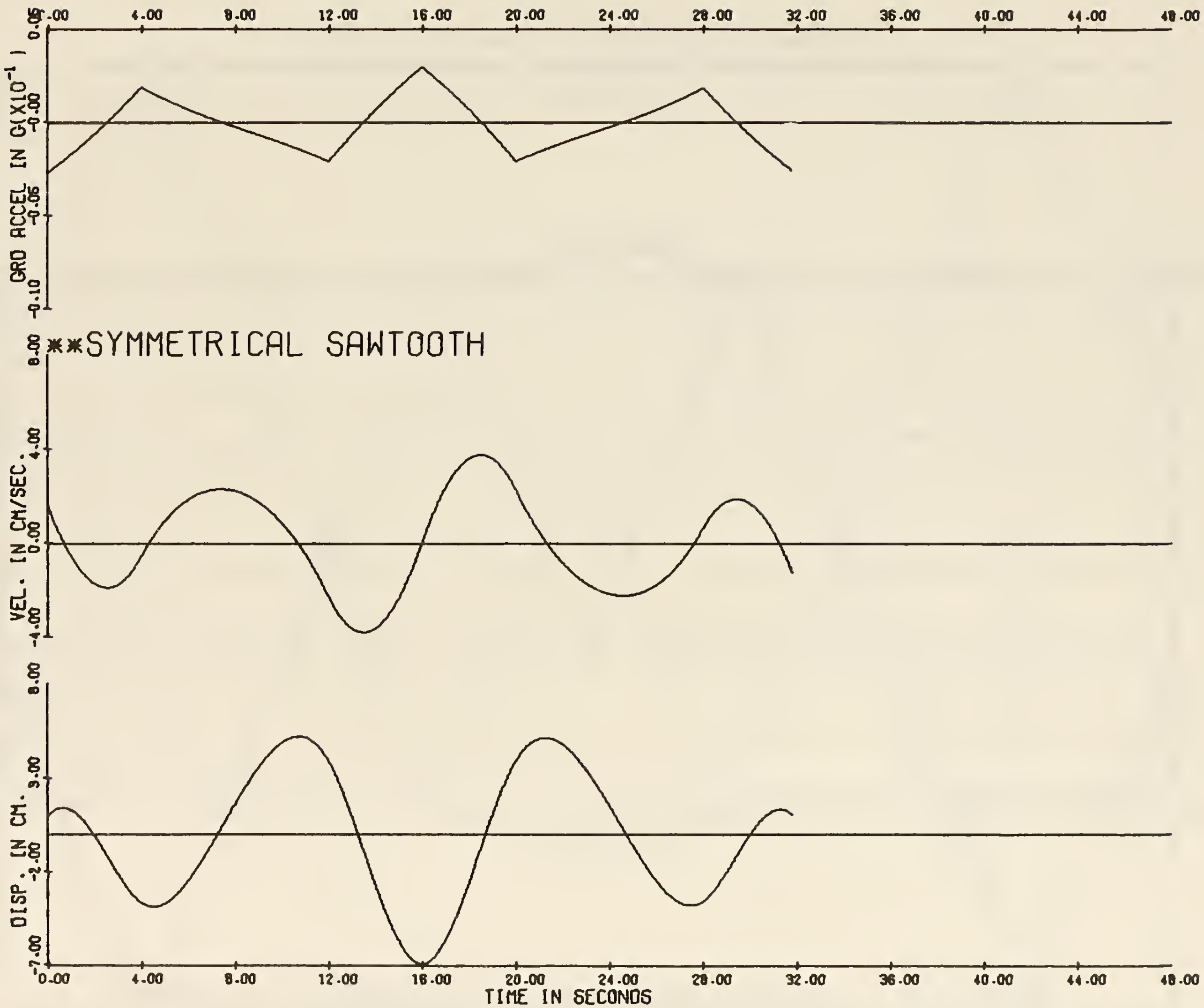


FIGURE 55 PHASE2 RESULTS FOR SYMMETRICAL SAWTOOTH ACCELERATION - CAL. TECH. VERSION

56-A

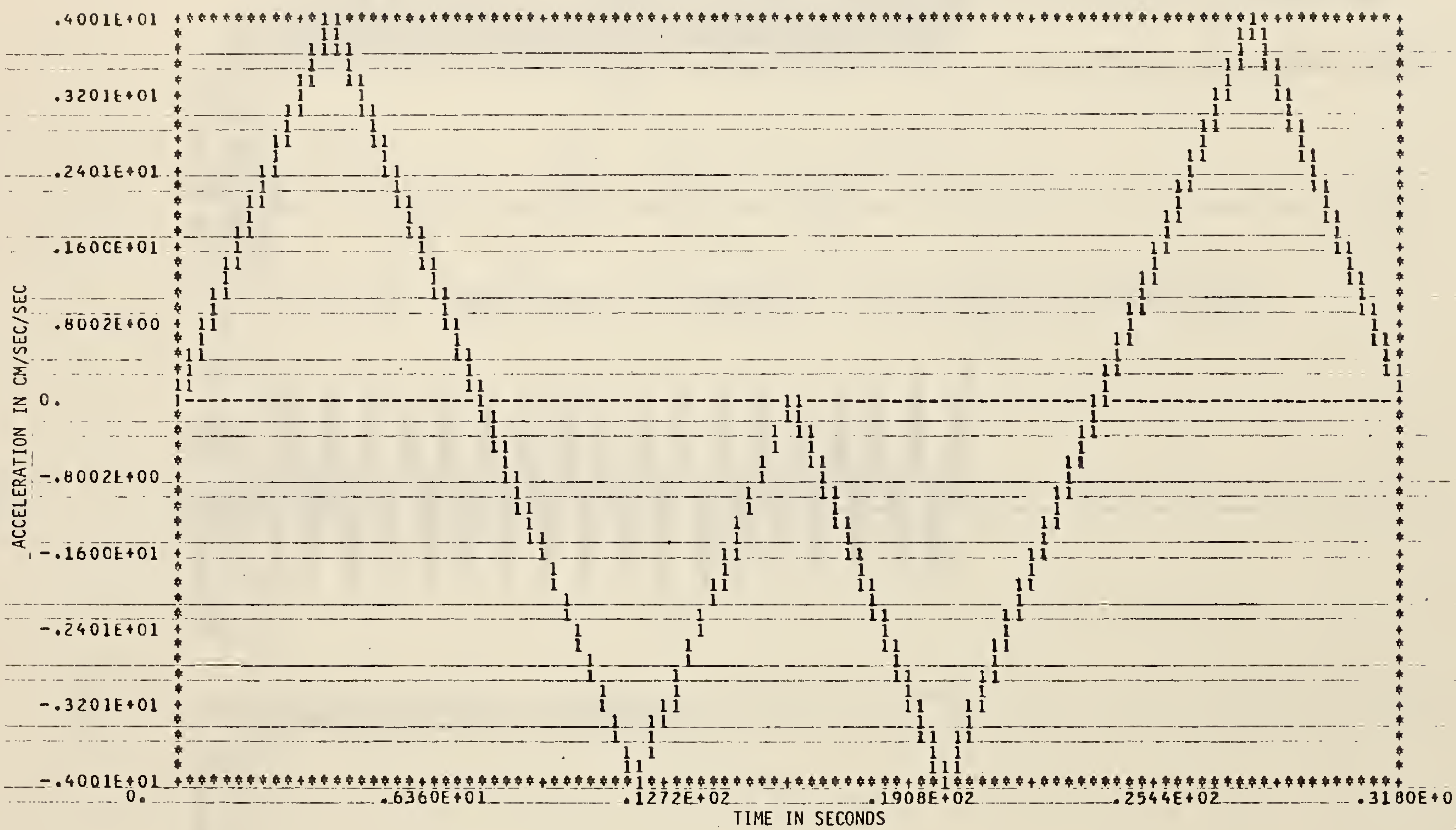


FIGURE 56-a PRINTER PLOT OF SYMMETRICAL SAWTOOTH ACCELERATION - ACCELERATION PRIOR TO HOLOWAY FILTERING

56-B

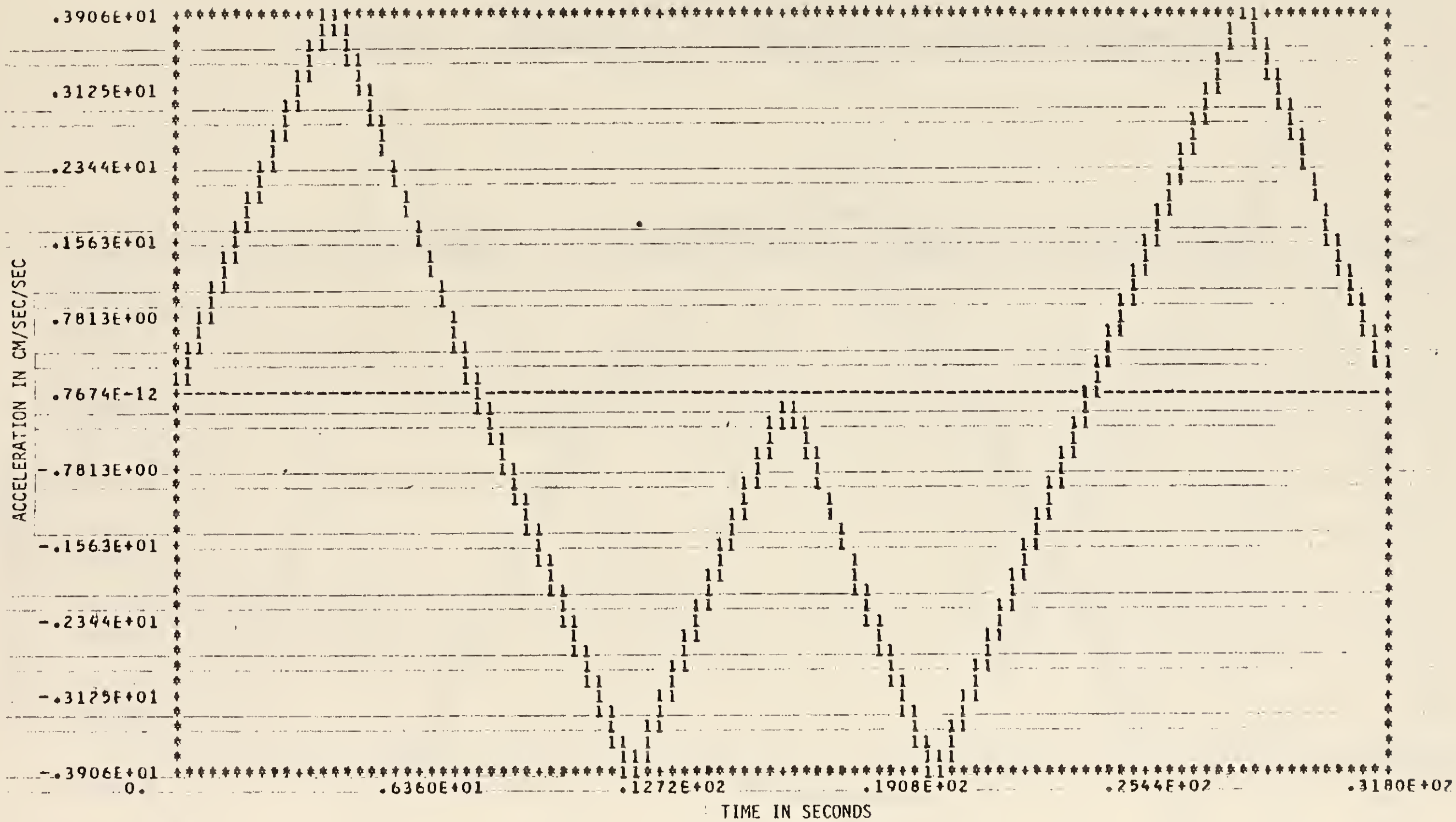


FIGURE 56-b PRINTER PLOT OF SYMMETRICAL SAWTOOTH ACCELERATION
ACCELERATION AFTER HOLOWAY FILTERING

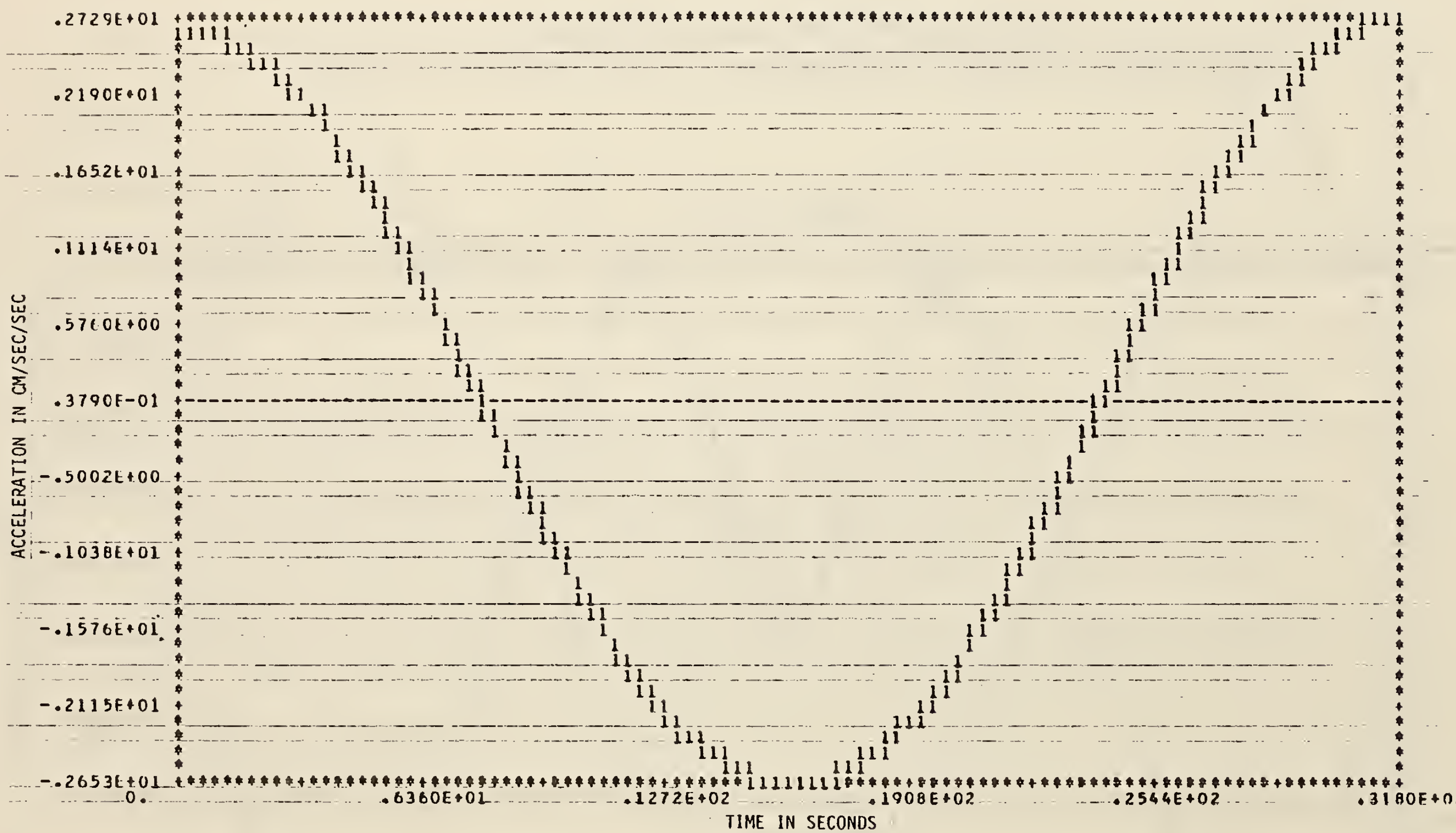


FIGURE 56-c PRINTER PLOT OF SYMMETRICAL SAWTOOTH ACCELERATION - ORMSBY FILTER OF ACCELERATION

56-c

σ-25

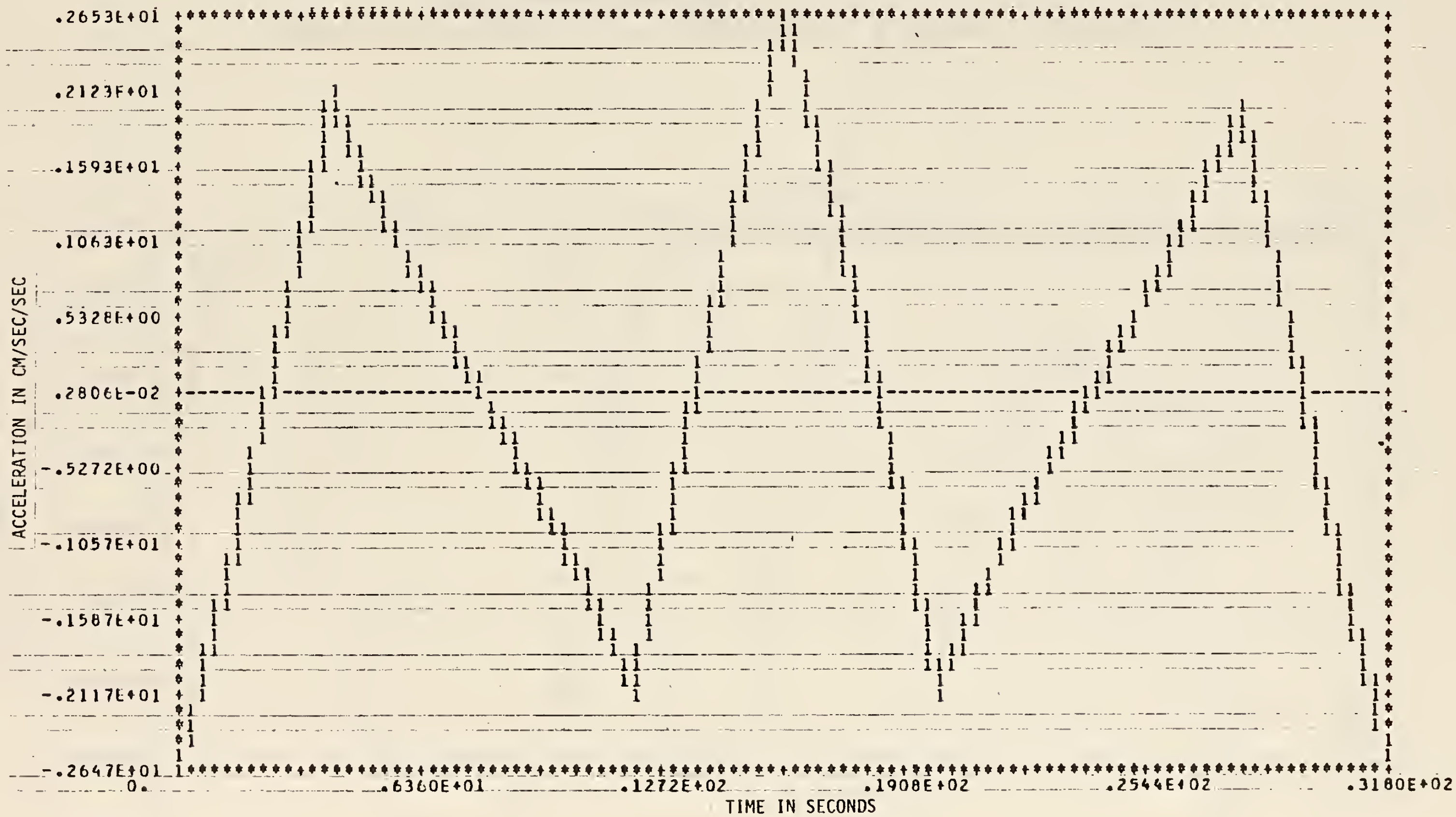


FIGURE 56-d PRINTER PLOT OF SYMMETRICAL SAWTOOTH ACCELERATION - ACCELERATION AFTER ORMSBY FILTERING

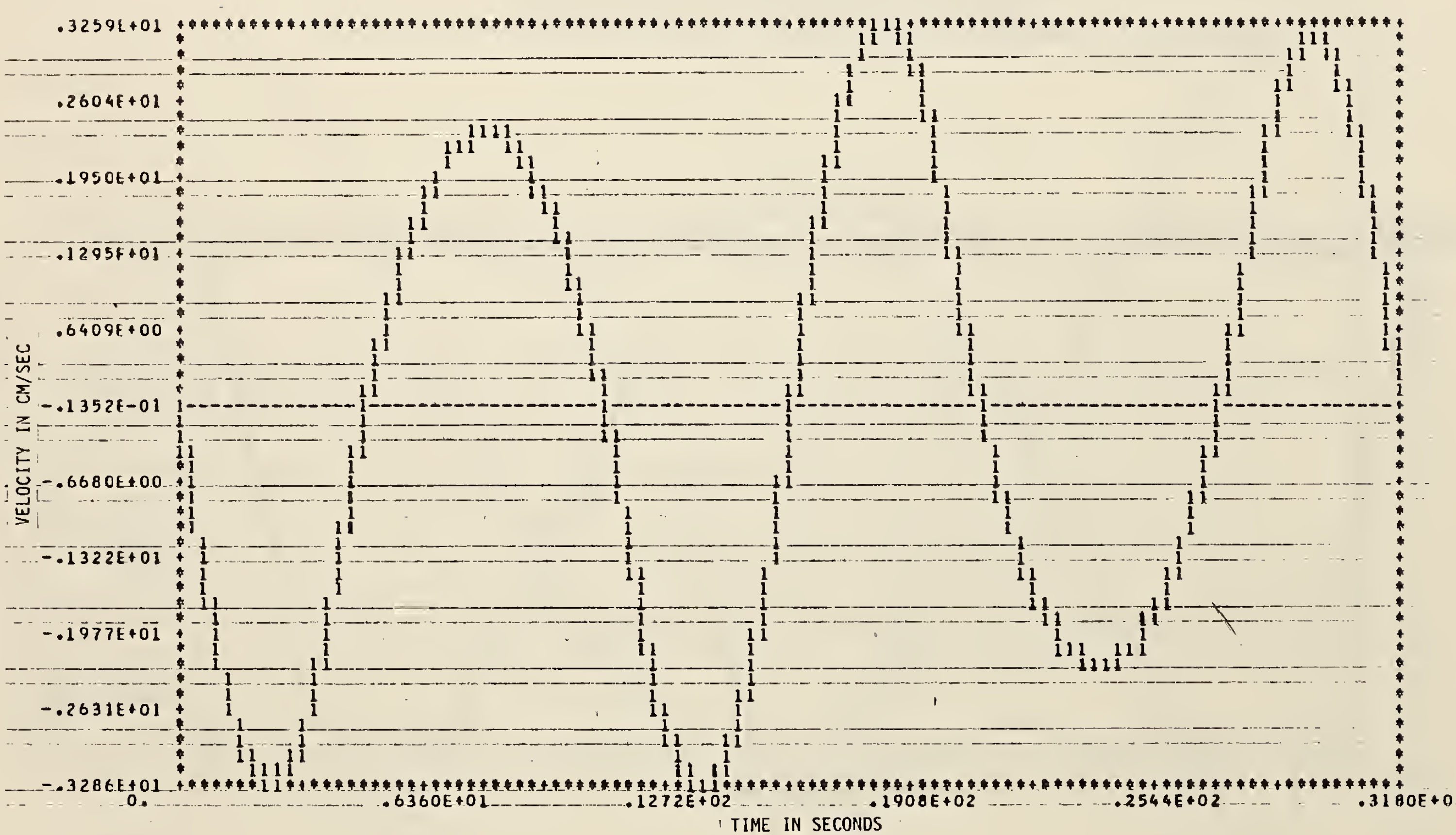


FIGURE 56-e PRINTER PLOT OF SYMMETRICAL SAWTOOTH ACCELERATION - INTEGRATED VELOCITY

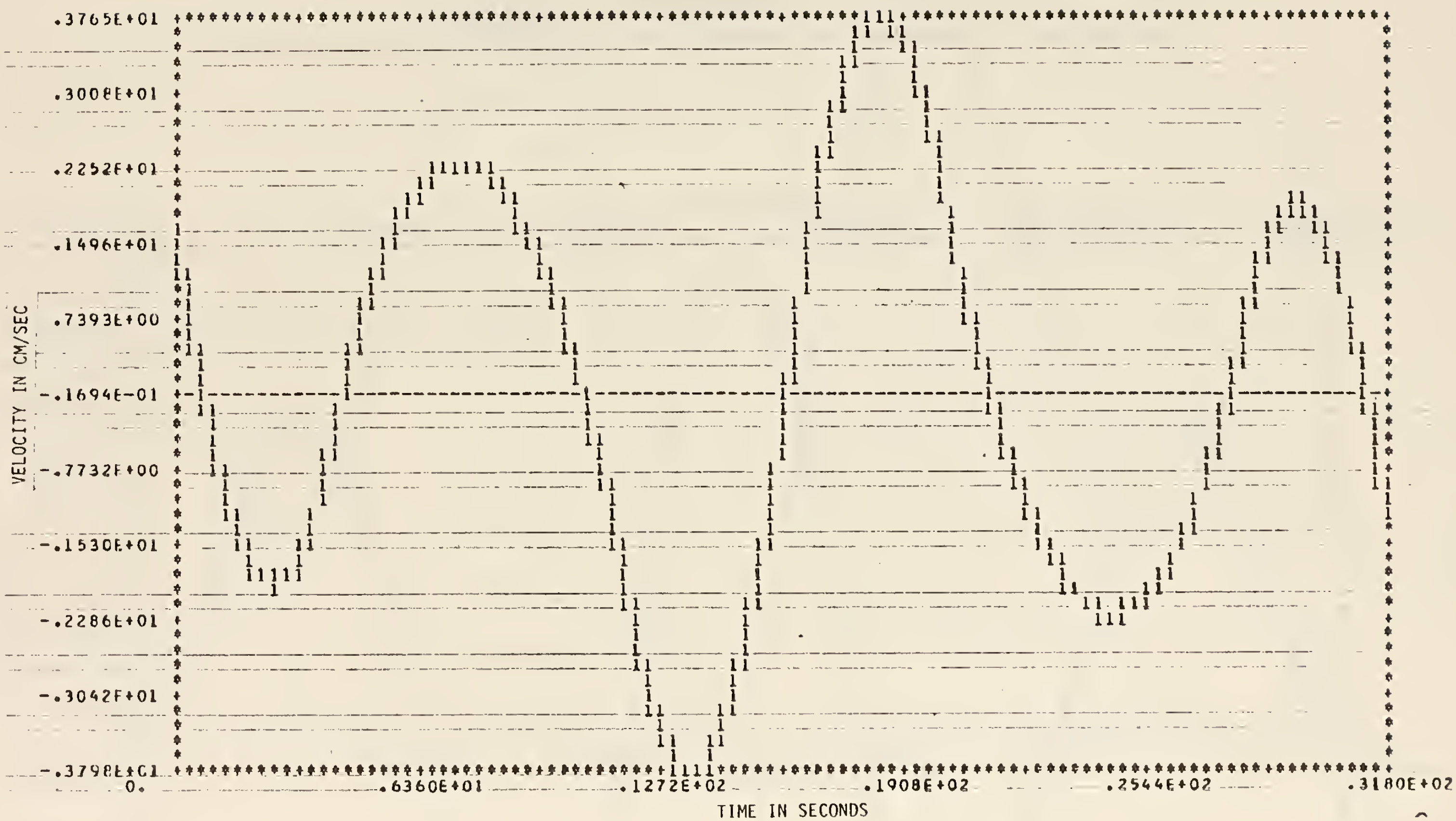


FIGURE 56-f PRINTER PLOT OF SYMMETRICAL SAWTOOTH ACCELERATION - FINAL VELOCITY AFTER FILTERING

56 g

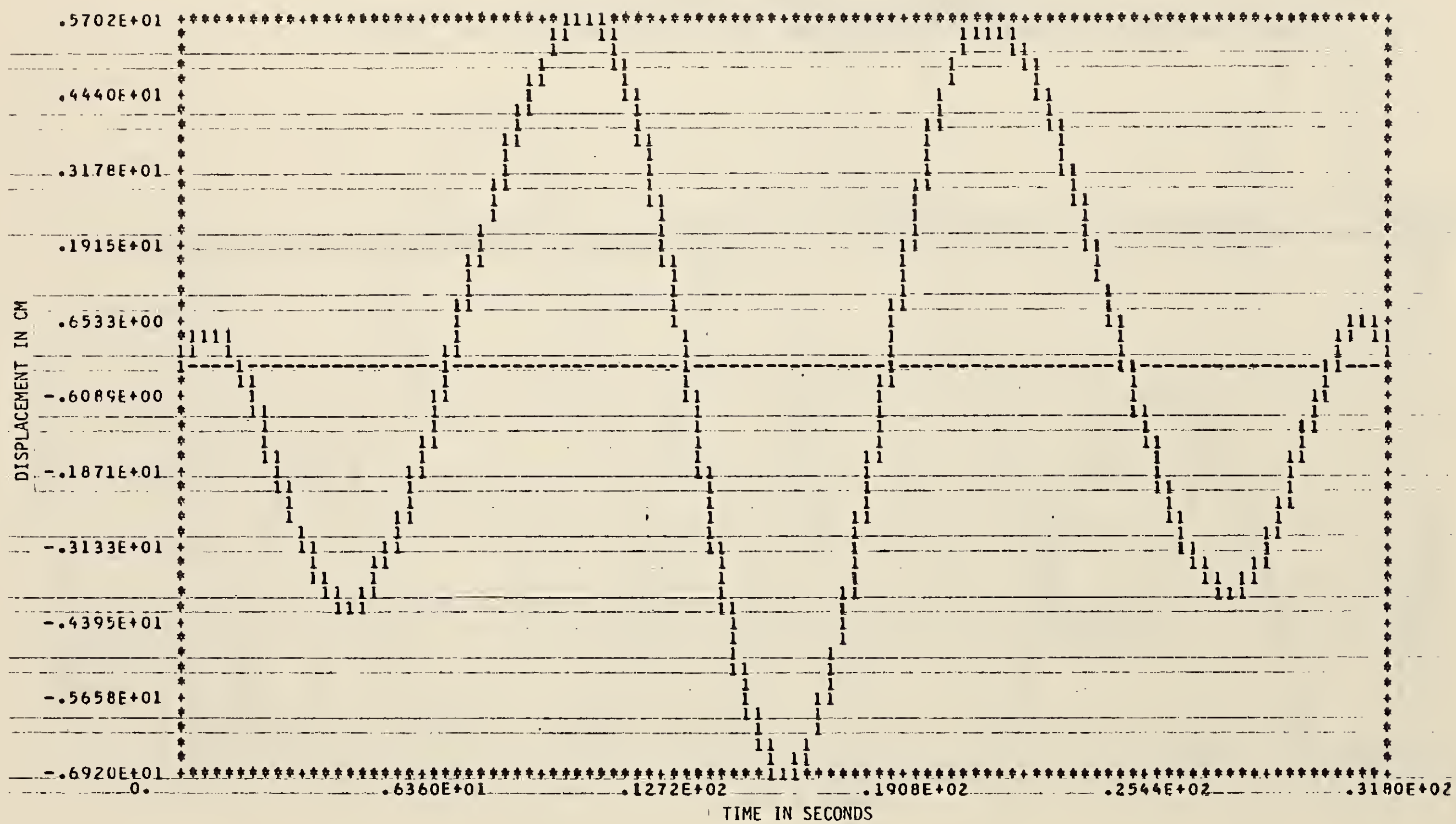


FIGURE 56-g PRINTER PLOT OF SYMMETRICAL SAWTOOTH ACCELERATION INTEGRATED DISPLACEMENT

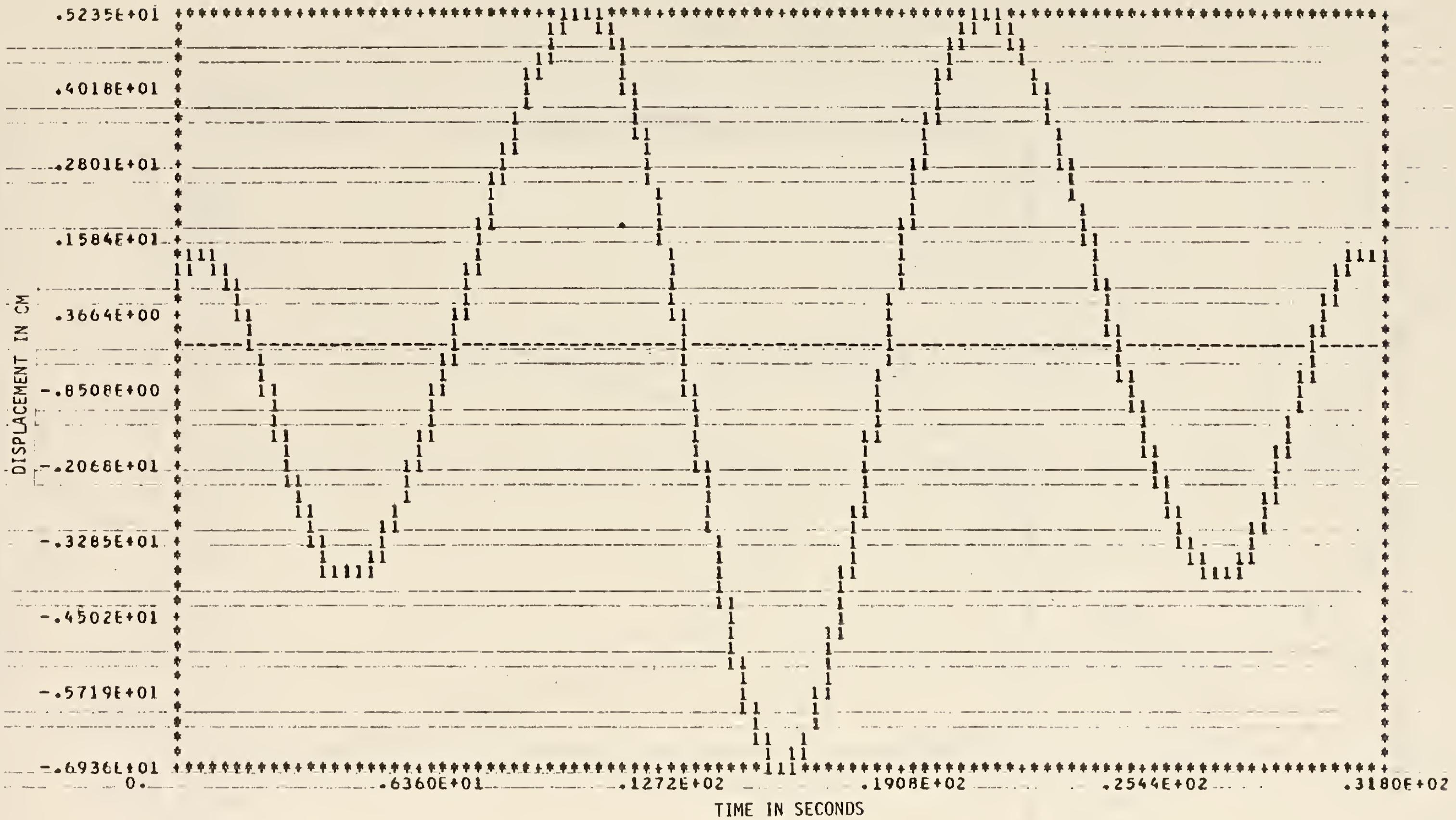


FIGURE 56-h PRINTER PLOT OF SYMMETRICAL SAWTOOTH ACCELERATION – FINAL DISPLACEMENT AFTER HIGH-PASS FILTERING AT 1/16 HZ

H-75

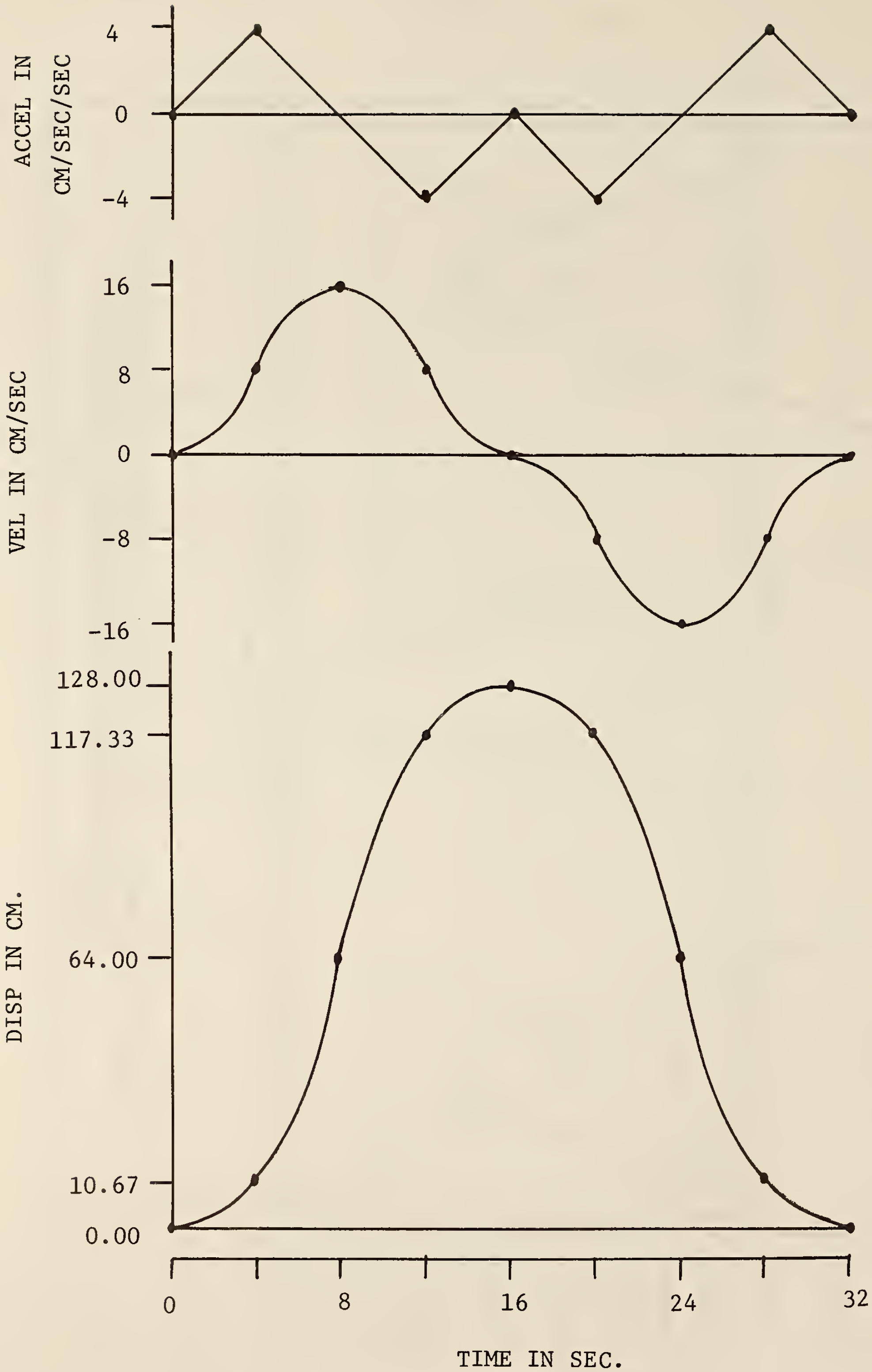


FIGURE 57 CALCULUS INTEGRATION TO VELOCITY AND DISPLACEMENT FROM SYMMETRICAL SAWTOOTH ACCELERATION CURVE

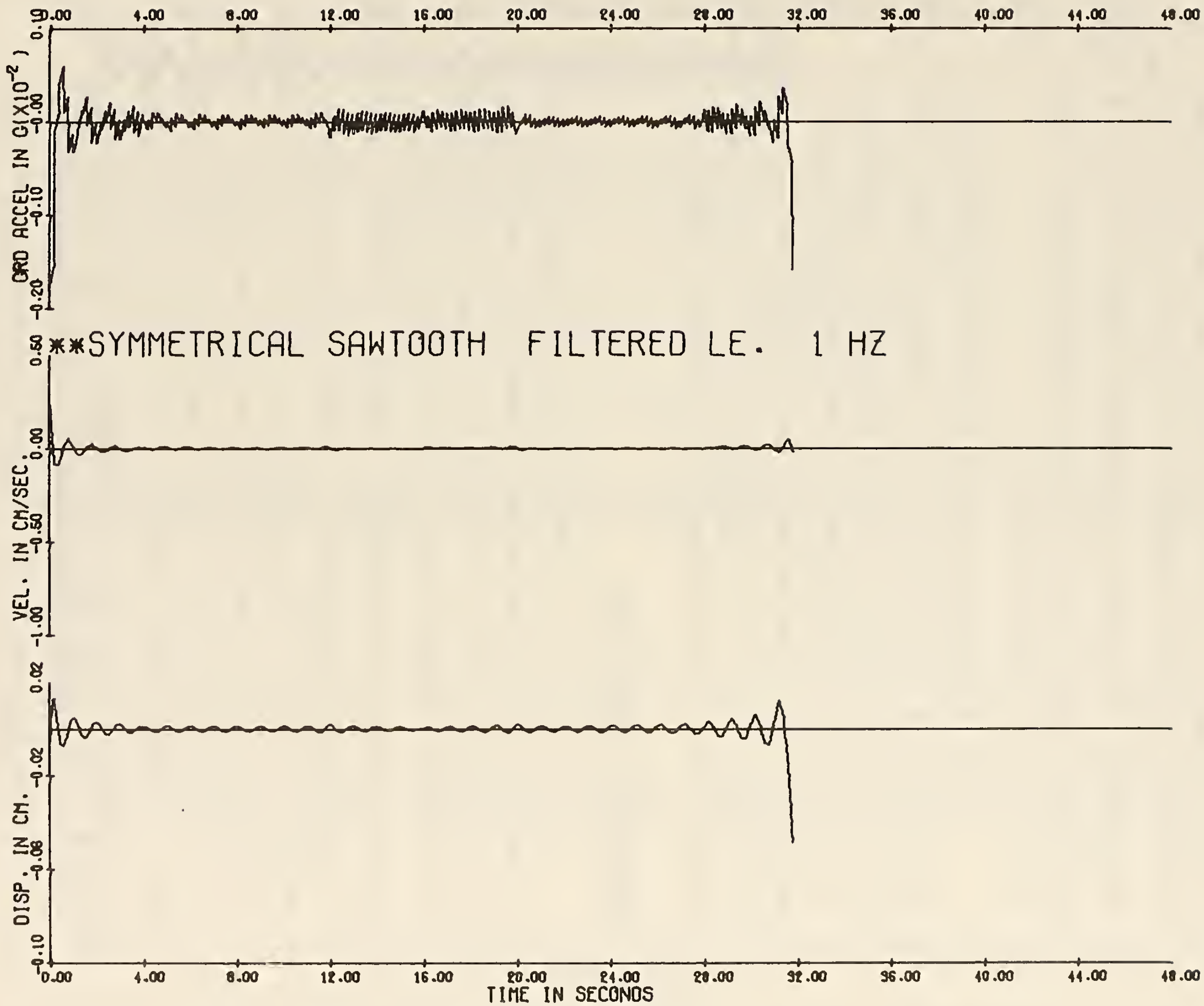


FIGURE 58 PHASE2 RESULTS FOR SYMMETRICAL SAWTOOTH - N4WAY = 10 VERSION

since these curves have periods longer than 10 seconds, Figure 56-h. In fact, the printer plots show identical scenes in Figures 56-a through 56-e; and then the Ormsby filtering on velocity reduces the remaining plots to a low noise level. This time, as expected, Figure 58 shows a fairly symmetrical noise curve about the centerline point since the input was arranged that way.

One conclusion we can draw from this example is that *when long periods are in the frequency range of the error frequency band, it may be impossible to guarantee the accuracy of the integrated time-displacement curve.* Recall that the long period errors of Figure 50 were put into the program by the leastsquare procedure, Holoway and Ormsby filters; they were not part of the original record.

9. Composition Input Data

To check the accuracy of the leastsquare procedure a composition input curve was prepared by superimposing a 3 Hz sine curve with a 20 Hz sine curve and a random number distribution together with a 1/10 radian/sec sloping line. Figure 59 shows the integrated results, and amply demonstrates the leastsquaring procedure is doing its job. Figure 60 used only the same random numbers as input, with a resultant time-displacement similar to Figure 59. Thereafter, to make sure no such frequencies could occur in the random number system, the acceleration record was first high-pass filtered at 2 Hz, Figure 61, and then integrated. It is interesting to note that the shape of the time-displacement history is similar for Figures 59, 60, and 61; verifying that the the displacements are in error and developed by the original program methodology.

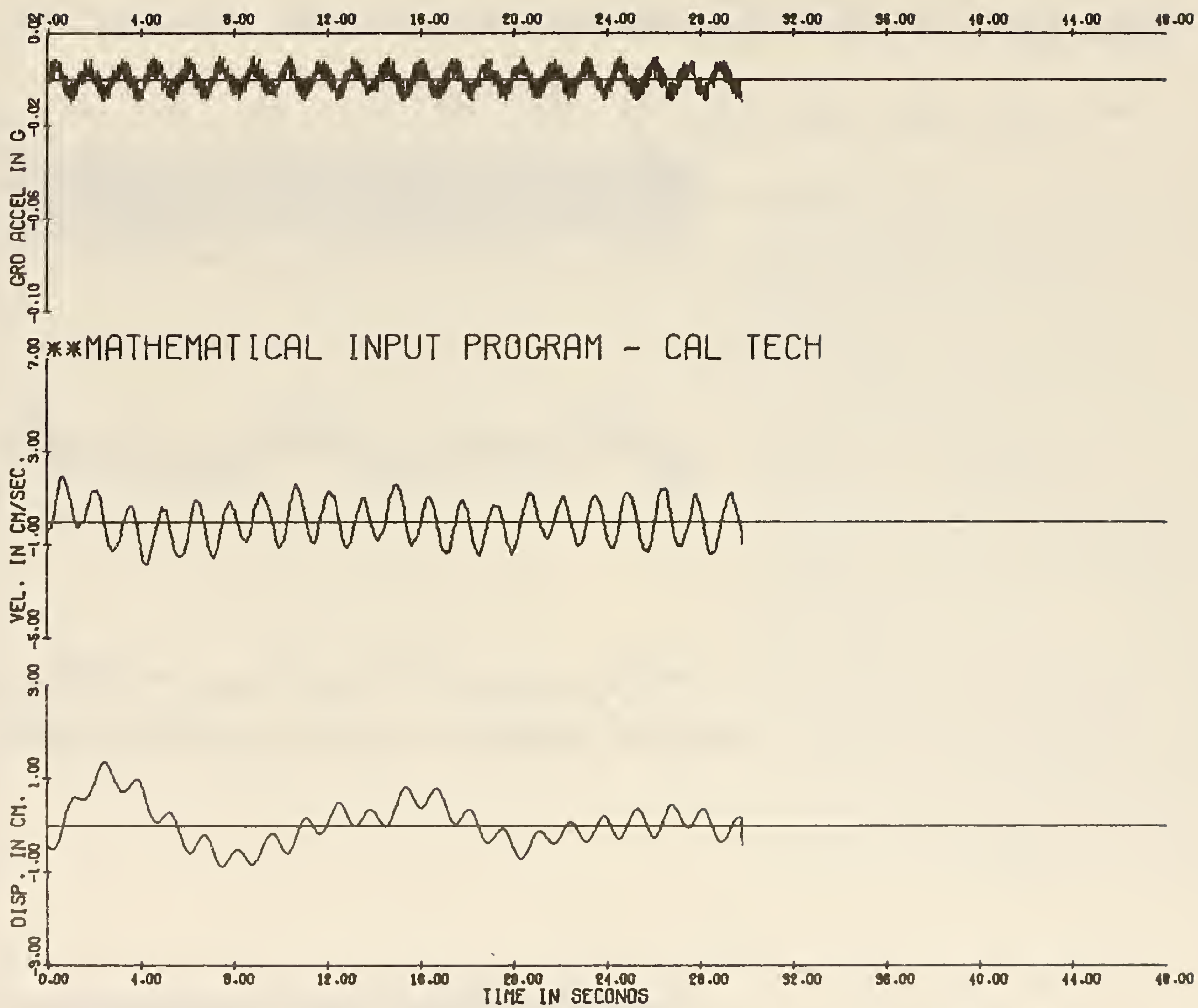


FIGURE 59 PHASE2 RESULTS FOR 3 HZ + 20 HZ + RANDOM NUMBERS + 1/10 RADIAN SLOPE ACCELERATION CURVE – CAL. TECH. VERSION

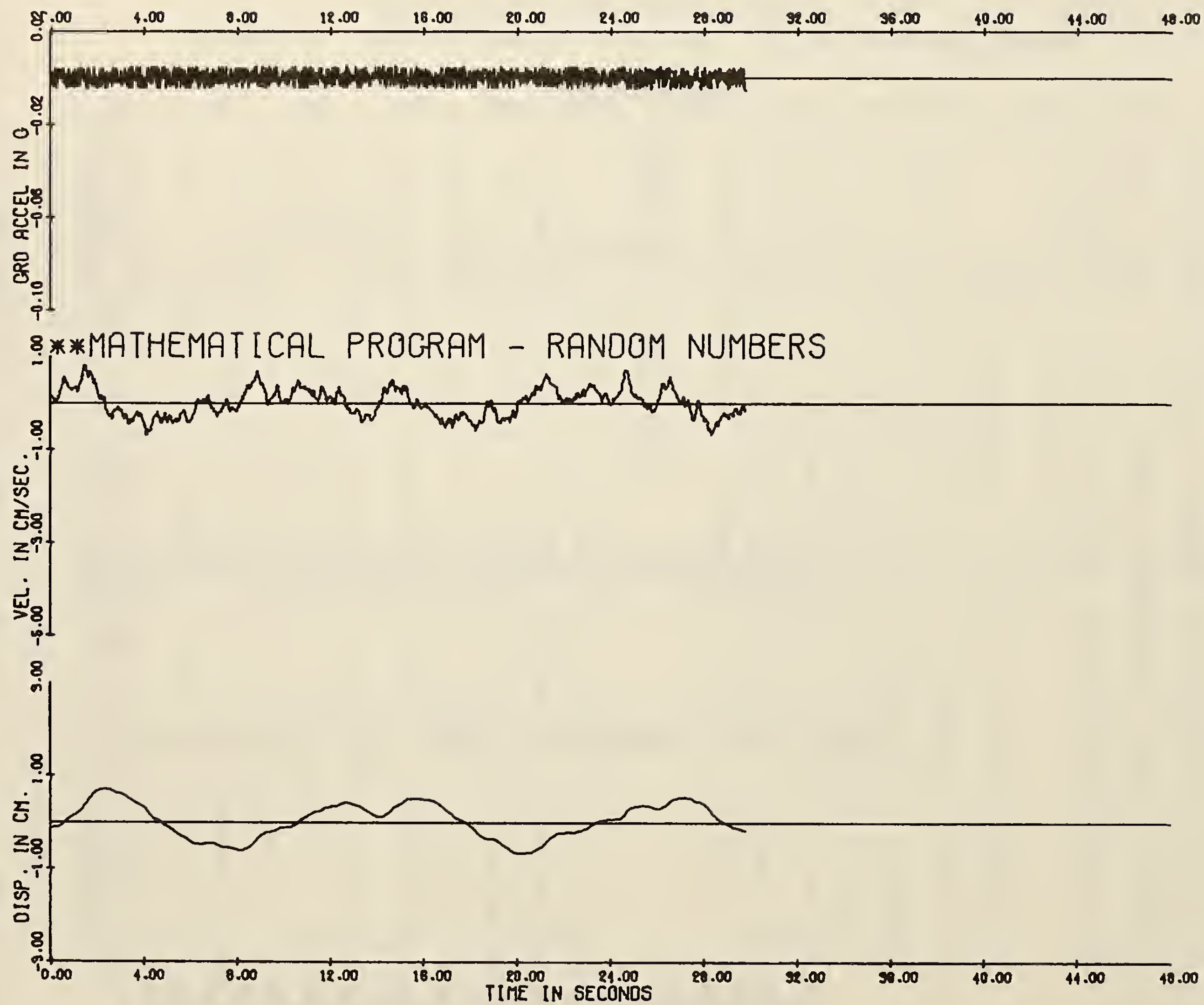


FIGURE 60 PHASE2 RESULTS FOR RANDOM NUMBERS ALONE - CAL. TECH. VERSION

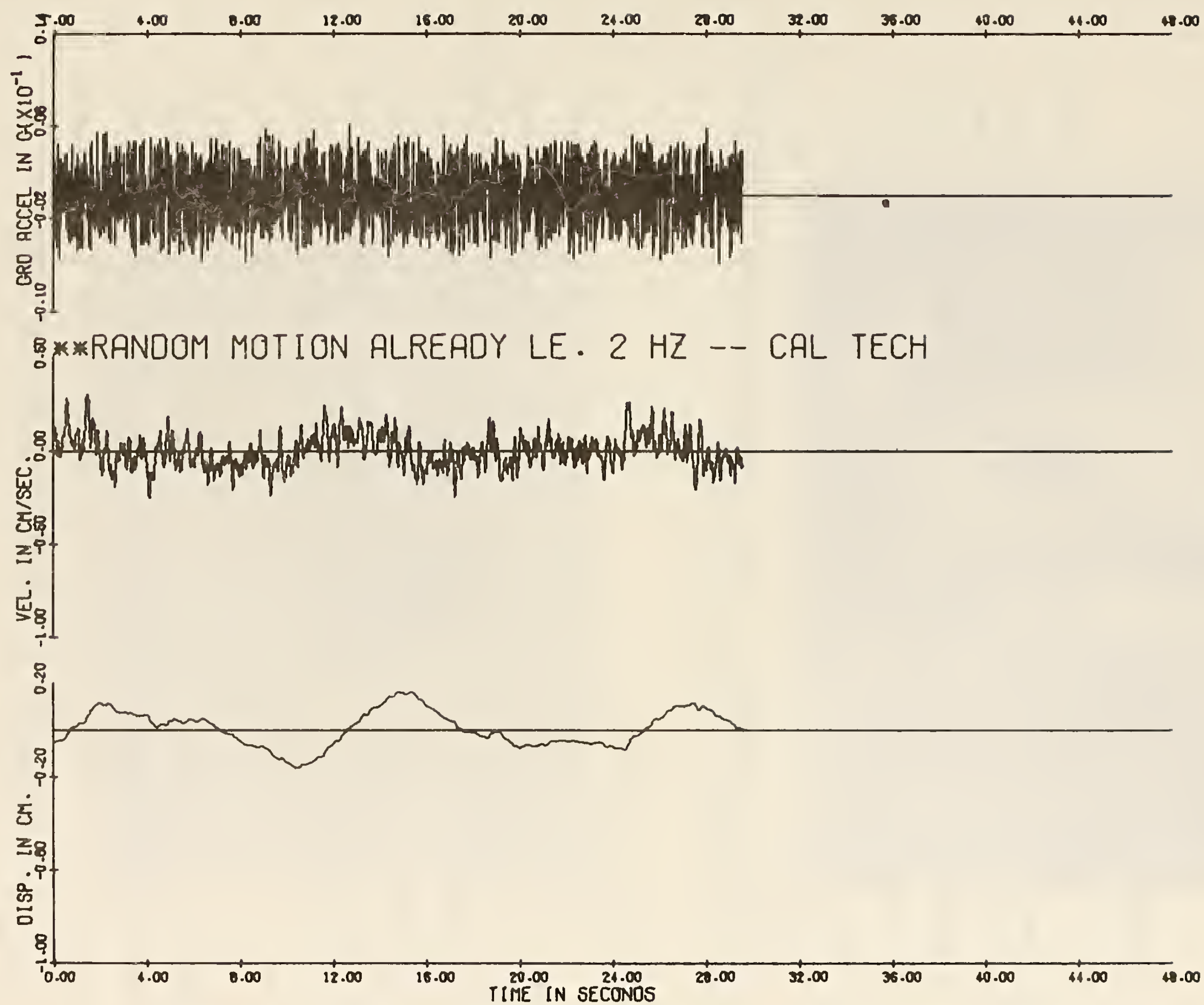


FIGURE 61 PHASE2 RESULTS FOR RANDOM NUMBERS ALREADY HIGH-PASS FILTERED AT 2 HZ - CAL. TECH. VERSION

VII. EXPERIMENTATION

1. Introduction

In this chapter it will be demonstrated that the Cal. Tech. program, with a suitable change in the Ormsby filter frequency, can produce a time-displacement curve that will accurately match a measured time-displacement history on the condition that the displacement curve oscillates about a zero baseline and has only frequencies higher than the filter frequency. This will put a restraint on the type of earthquake displacement records that can be predicted, i.e., those having no significant final displacement from the original position of the earth.

2. Free Swing Experiment

Prior to a random motion experiment it was decided to produce a free swing experiment for instrument calibration purposes. Figure 62-a shows a pendulum device used to generate accelerations. A piezoelectric accelerometer was attached to the pendulum and connected to a Sanborn recorder to create the acceleration trace. To form the pendulum two vertical plates were attached to a top and bottom plate to form a parallelogram. The upper plate was fixed in a horizontal position. As the pendulum was rotated along its path, the bottom plate always remained in a horizontal position regardless of the angle of the pendulum. A Model 818 Piezotron Accelerometer*, Figure 62-b, was rigidly attached to the horizontal plate to sense accelerations normal to the plate; however, only the vertical components of acceleration were recorded.

---

* Manufactured by Kistler Instruments Company, Overlake Industrial Park, Redmond, Washington 98052.

FIGURE 62-a PENDULUM DEVICE USED TO GENERATE ACCELERATIONS

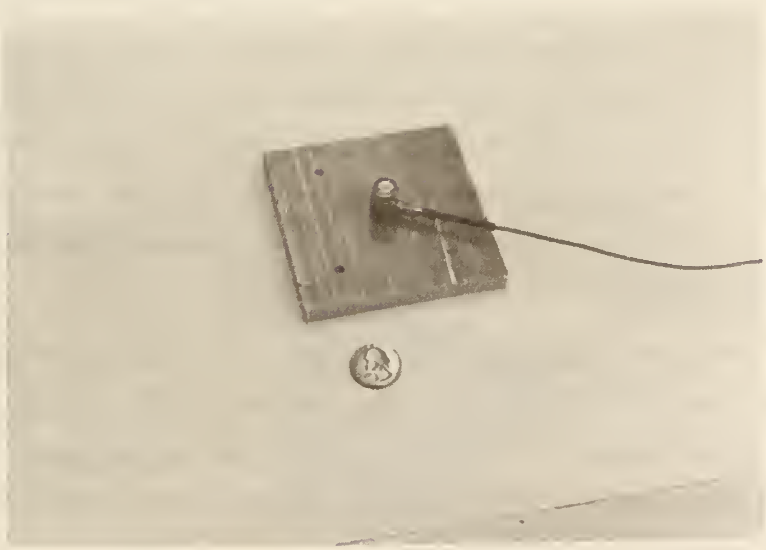


FIGURE 62-b MODEL 818 PIEZOTRON ACCELEROMETER

The piezotron accelerometer contained a compression-type piezoelectric sensing element, Figure 63, which included a seismic mass and stacked, parallel-connected quartz crystals (or plates), which were assembled under controlled preload pressure into the preload sleeve. The sensing element was mechanically isolated from the housing, preventing mounting strain from causing either spurious signals or zero shift. The mounted resonant frequency was 31.5 KHz, well above the range of strong motion instruments. The frequency response was from 0.8 Hz to 5000 Hz (within ± 6%) and had a zero voltage deviation in the frequency range of 7.5 to 900 Hz at $+75^{o}$ (see Figure 64).

To calibrate the accelerometer system, the pendulum was dropped in a single free swing and the accelerometer was caused to pass between two flashlights and photoelectric cells, as shown in Figures 62-a and 65-a. The photoelectric cells were connected to a voltage amplifier (see the circuit diagram of Figure 65-b) which was then connected to a Tectronic Type 551 Dual Beam Oscillograph with Type 53A plug-in units for voltage response. The dual traces were then recorded on polaroid film, Figure 66, showing blips when the accelerometer interrupted the flashlight beam to its corresponding cell.

From mechanics it is known that the radial component of acceleration is $\frac{v^2}{r}$. At the bottom swing of the pendulum the radial acceleration is all that exists and is directed upward being normal to the bottom plate. From Figure 66-a the following calculations are made:

Sweep at 0.02 sec/cm on the screen

Correction of 1.1 by timing with a stopwatch makes the sweep speed 0.02(1.1) = .022 sec/cm.

Distance between blips = 1.25 cm on the screen.

Distance between photocells = 10 cm.

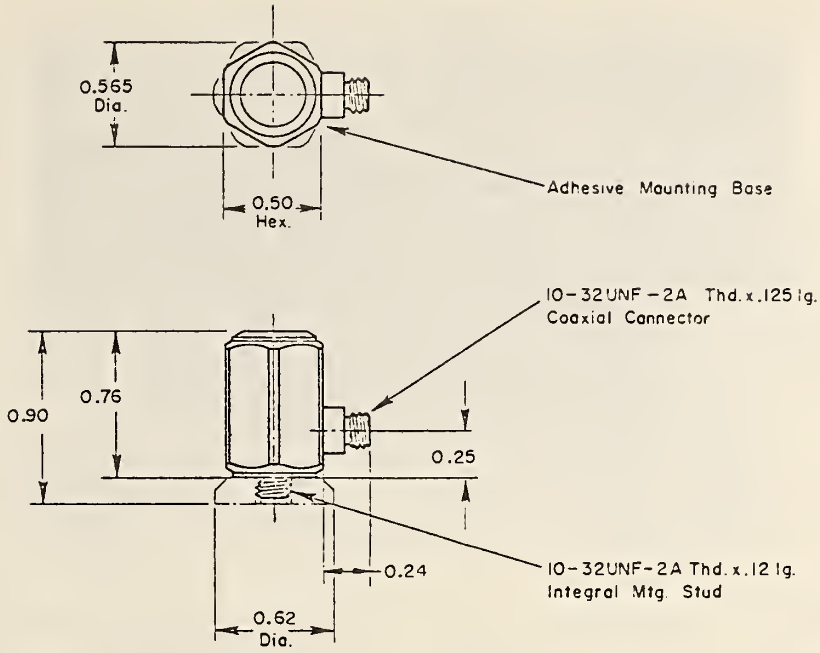


FIGURE 63-a BASIC DIMENSIONS OF MODEL 818 PIEZOTRON ACCELEROMETER

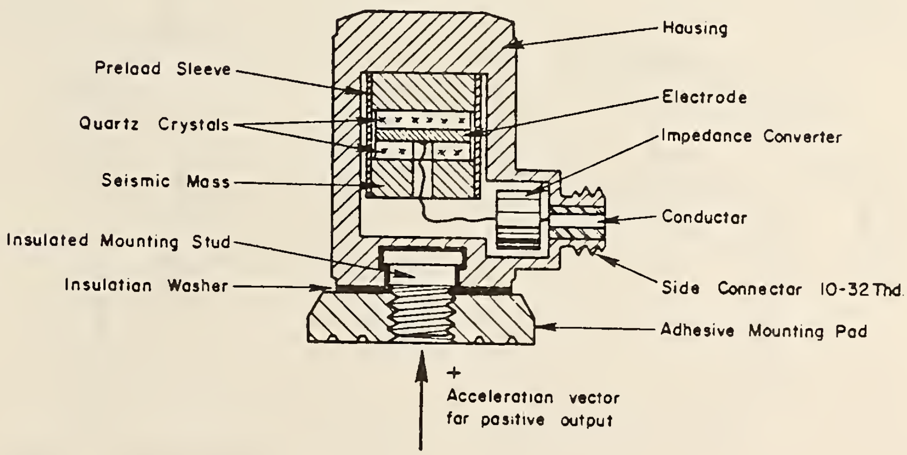


FIGURE 63-b CROSS SECTION OF PIEZOTRON ACCELEROMETER

TABLE IV.

LOW-FREQUENCY RESPONSE CHARACTERISTICS

Model	818	817
Time Constant / Attenuation	0.2 sec	2.0 sec
-1 percent	5.6 Hz	0.56 Hz
-2 percent	4.0 Hz	0.40 Hz
-5 percent	2.4 Hz	0.24 Hz
-10 percent	1.6 Hz	0.16 Hz
-30 percent	0.8 Hz	0.08 Hz

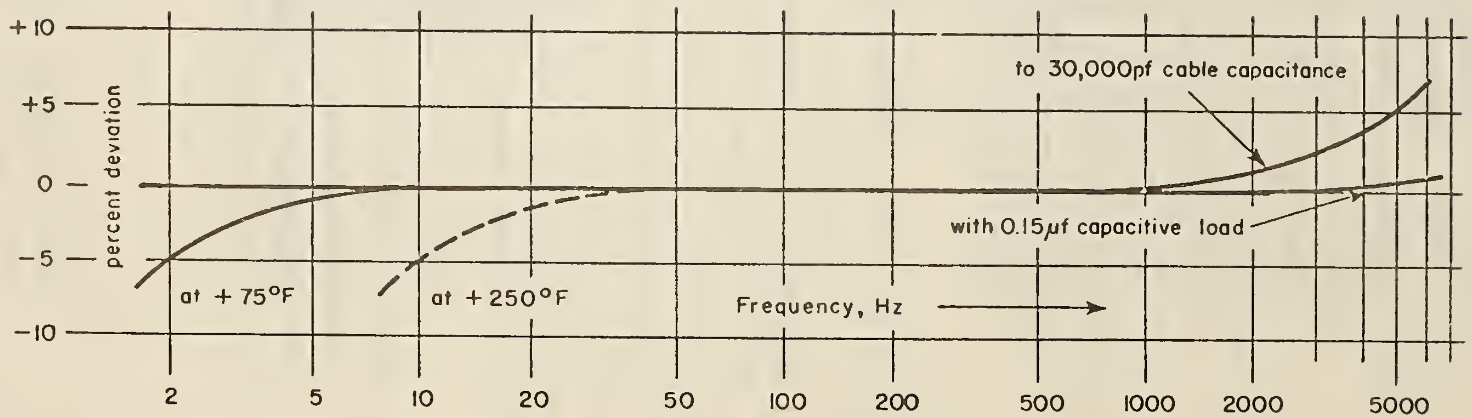


FIGURE 64 FREQUENCY-RESPONSE CHARACTERISTICS OF MODEL 818 AND 817 ACCELEROMETERS

FIGURE 65-a PANEL VIEW OF PHOTOELECTRIC CELL DEVICE TO MEASURE THE PENDULUM VELOCITY

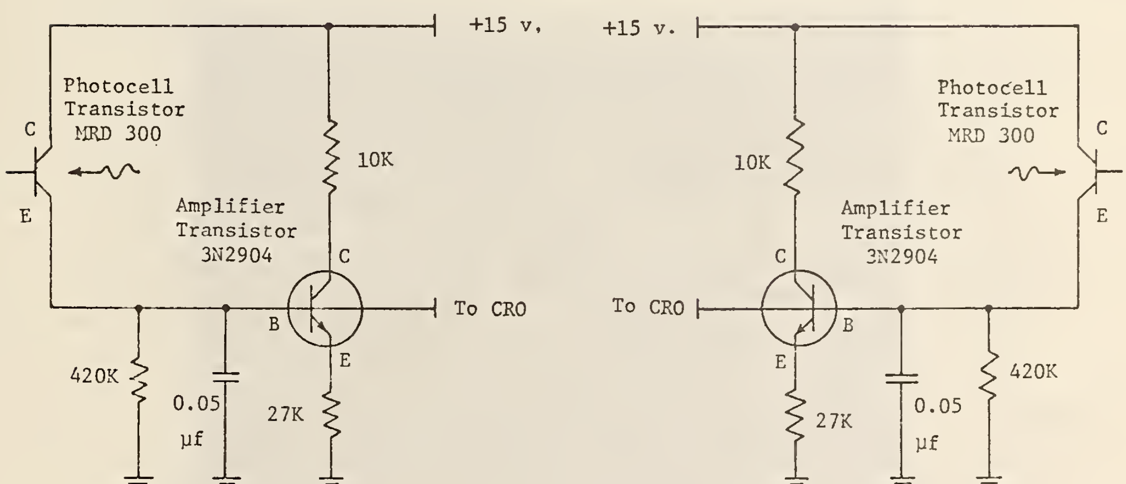
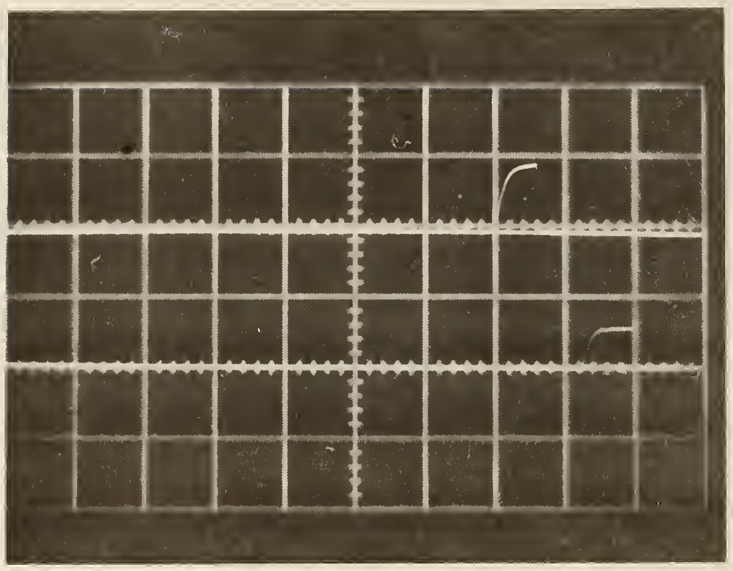
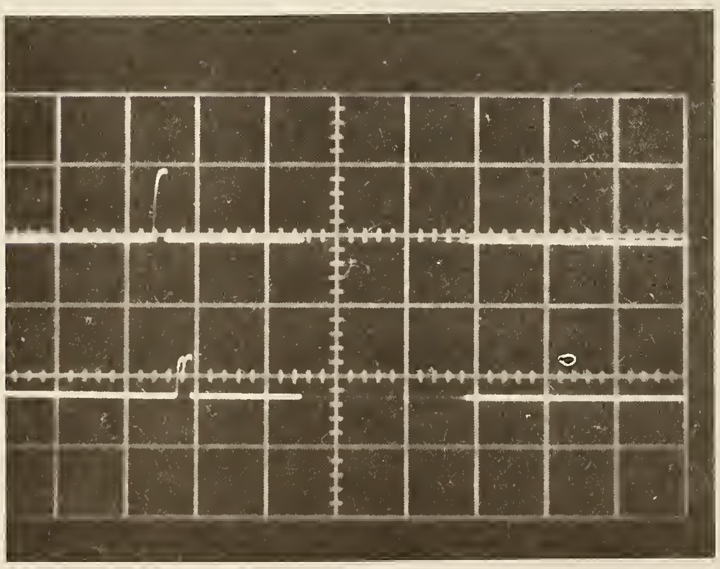


FIGURE 65-b CIRCUIT OF VOLTAGE AMPLIFIER FOR PHOTOCELL DEVICE



(a) RECORDING SPEED AT 0.02 SEC/CM



(b) RECORDING SPEED AT 0.10 SEC/CM

FIGURE 66 PHOTOGRAPHS SHOWING BLIPS ON DUAL BEAM TRACES AS PENDULUM PASSED IN FRONT OF PHOTOCELLS

$$v = \frac{10}{(1.25)(.022)} = 363.63 \text{ cm/sec.}$$

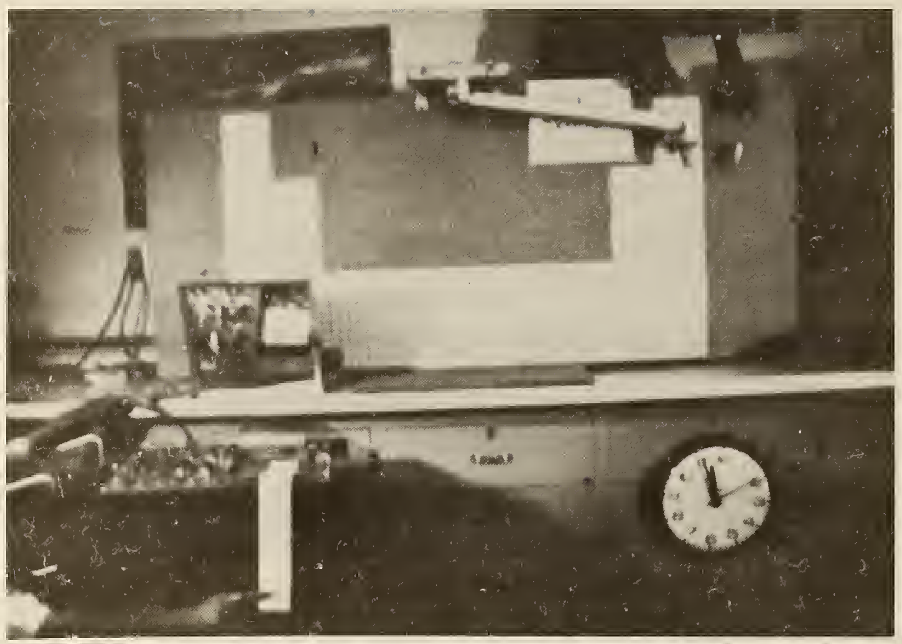
With a pendulum radius of 22.76" = 57.81 cm.

$$a = \frac{v^2}{r} = \frac{363.63^2}{57.81} = 2287.26 \text{ cm/sec/sec.}$$

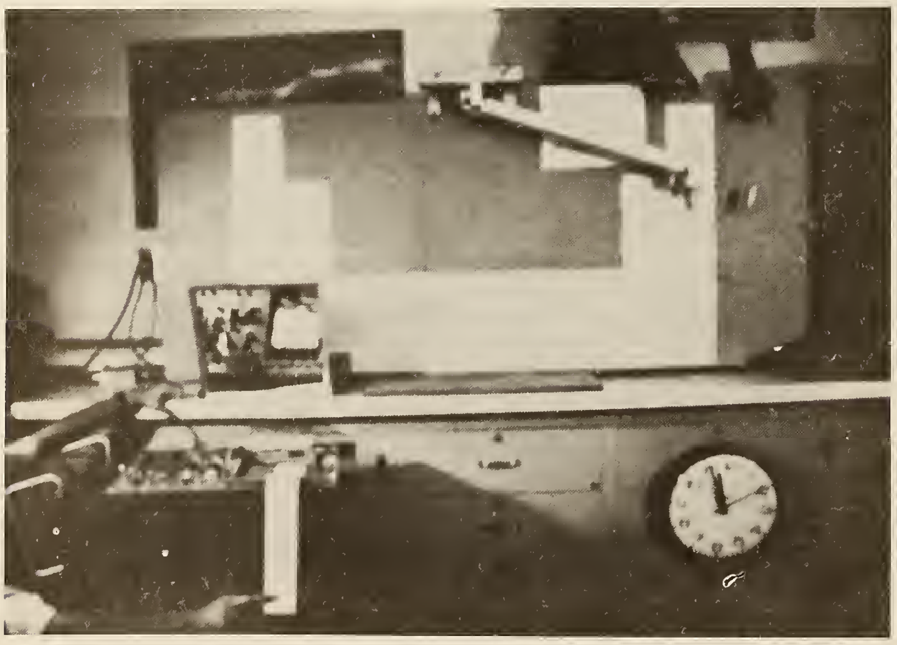
For Figure 66-b a = 1689.26 cm/sec/sec. For a sweep of 0.05 cm/sec (trace not shown) a = 2017.69 cm/sec/sec. An average value of 1998.07 cm/sec/sec could then be used to scale the lowest peak on the acceleration curve in the free swing experiment; however, the maximum value that can be developed from theoretical mechanics is 2G or 1961.33 cm/sec/sec, so that was used instead. This is on the basis that the acceleration at the beginning of the swing is G. Figure 67 shows four frames from a film record in a further free swing experiment that shows a free block dropped simultaneously with the pendulum keeping its relative position during at least the first half of the drop height and verifies that the initial acceleration was close to G.

In the free swing experiment the pendulum was initially set into motion from a horizontal position and allowed to swing back and forth for 5.58 seconds through approximately 7 cycles. While the pendulum was swinging, the accelerometer was sensing vertical accelerations which were recorded by means of a Sanborn recorder. The results appeared on the recording paper as a sinousoidal-like curve.

During the free swing experiment the action was recorded by a movie camera at the rate of 48 frames per second. A grid made of one-inch squares was placed in the background so that the relative vertical displacement of the pointer at the bottom of the horizontal plate could be retrieved from the film record. A microfische film reader was used to record the relative

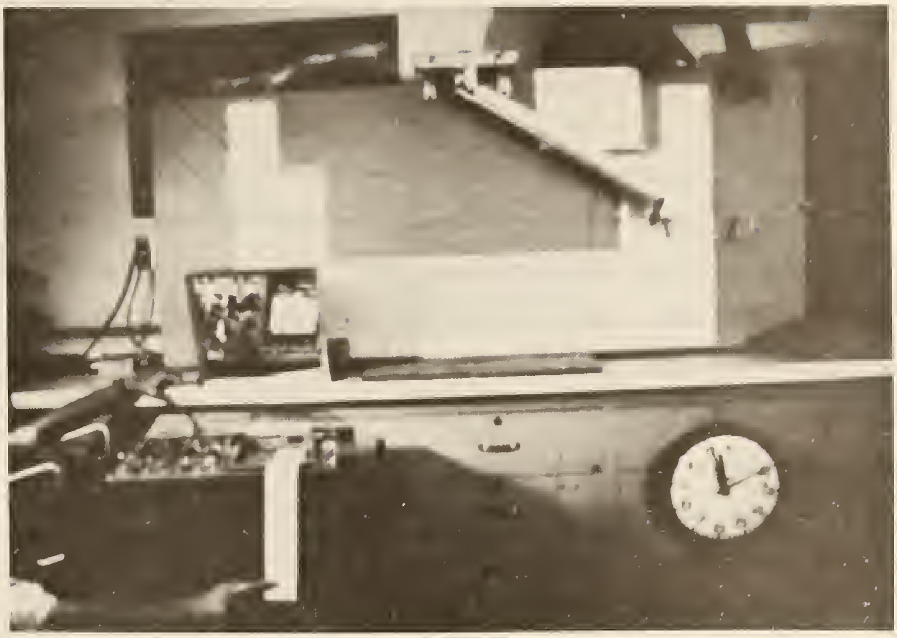


(a)

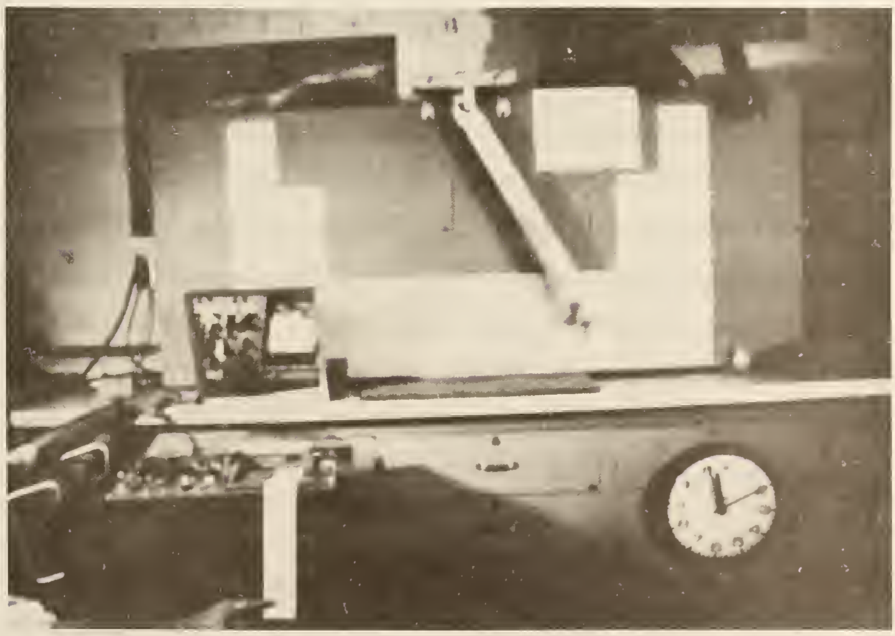


(b)

FIGURE 67 FOUR FRAMES FROM FREE SWING MOTION PICTURE RECORD



(c)



(d)

FIGURE 67 CONTINUED.

vertical position of the pointer by reading each frame. The record of actual vertical displacement is shown in Figure 68.

Meanwhile the accelerometer trace was digitized on the Electrak machine and processed through the PHASE1 program where a plot of the acceleration trace was prepared, Figure 69. The data was corrected to eliminate accidental digitization errors, scaled, baseline corrected, and then punched on computer cards for double integration in the PHASE2 program. An assessment was made of the beginning and end of the acceleration record and the portion shown in Figure 70 was first used for integration.

Figure 71 shows the integration via the original Cal. Tech. program. With the displacement data approximating at 2 Hz variation the Ormsby filter was next changed to 1 Hz (N4WAY = 10) and the free swing data once again integrated, Figure 72. When we compare the displacement results to those of Figure 73, we observe a very good match, *provided the actual displacement record is also filtered so that oscillations occur about a new baseline where the initial and final displacements are zero.* In Figure 73 the PHASE2 program was used to high-pass filter the input displacement data at 1 Hz by letting N2WAY = 8 and N4WAY = 10. The input displacement data was then placed in the acceleration array for plotting purposes only (the acceleration axis is ignored) while the filtered displacement was printed correctly at the bottom of the figure.

To ascertain if the length of acceleration record used might influence the displacement results both end extensions

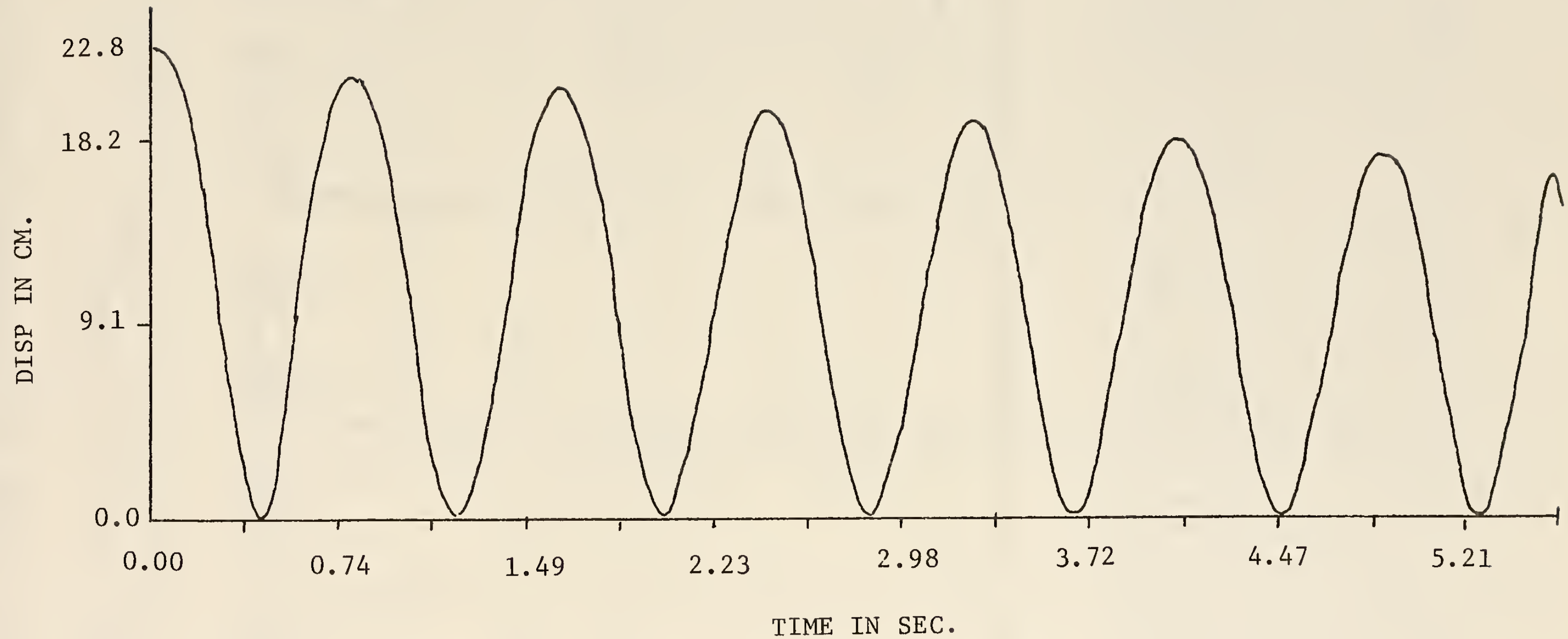


FIGURE 68 PLOT OF ACTUAL DISPLACEMENT OF FREE SWING EXPERIMENT

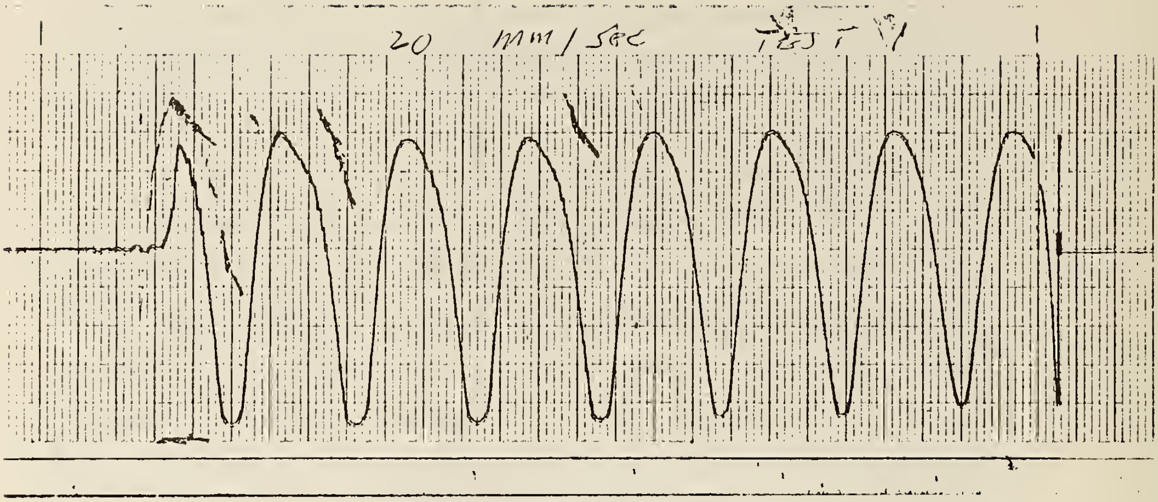


FIGURE 69 FREE SWING ACCELERATION TRACE ON SANBORN PAPER

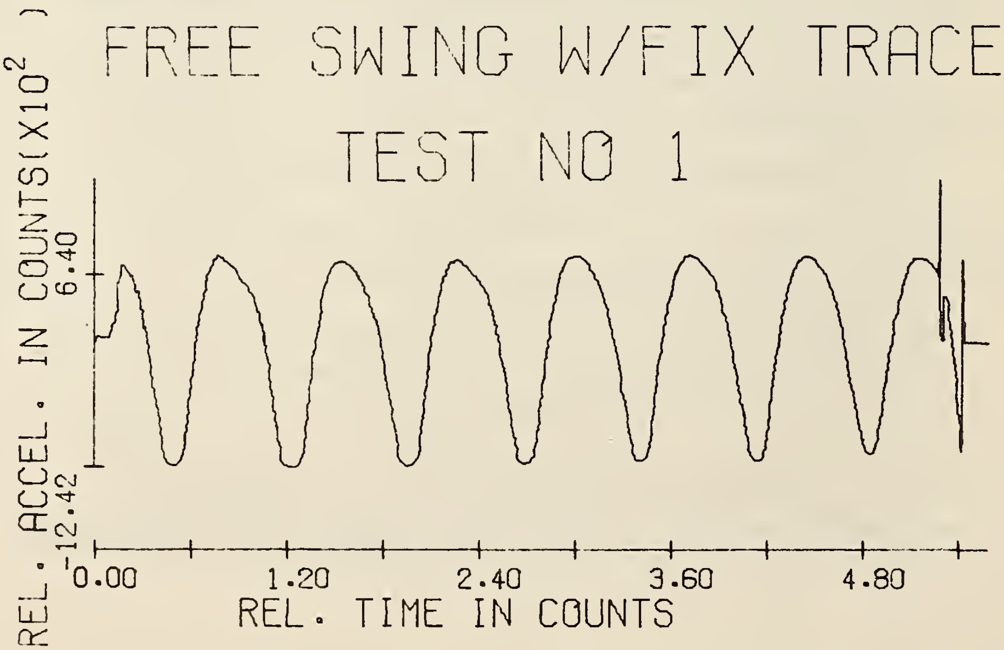


FIGURE 70 PART OF FREE SWING ACCELERATION RECORD BASELINE CORRECTED VIA PHASE1

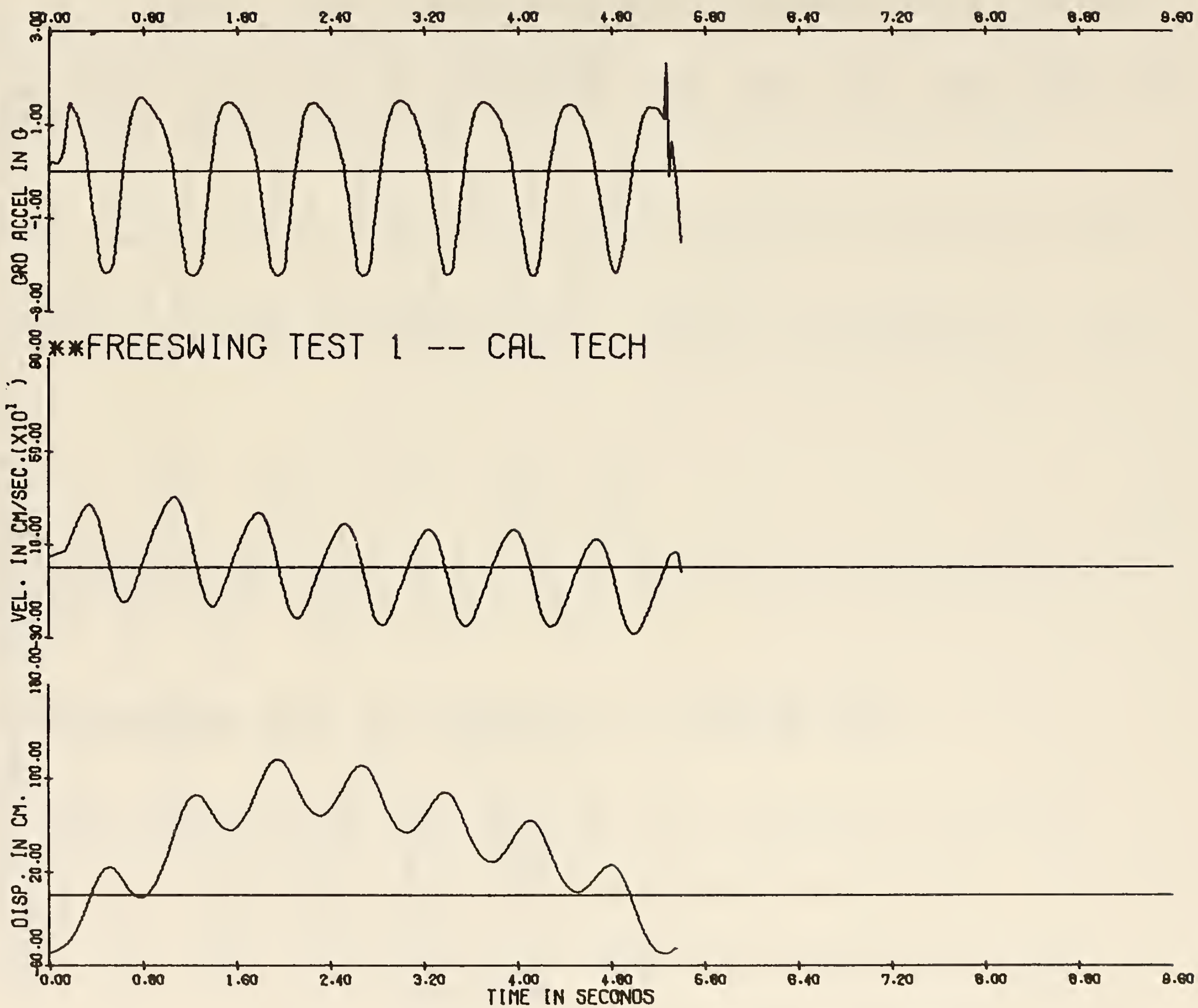


FIGURE 71 PHASE2 RESULTS OF FREE SWING RECORD - CAL. TECH. VERSION

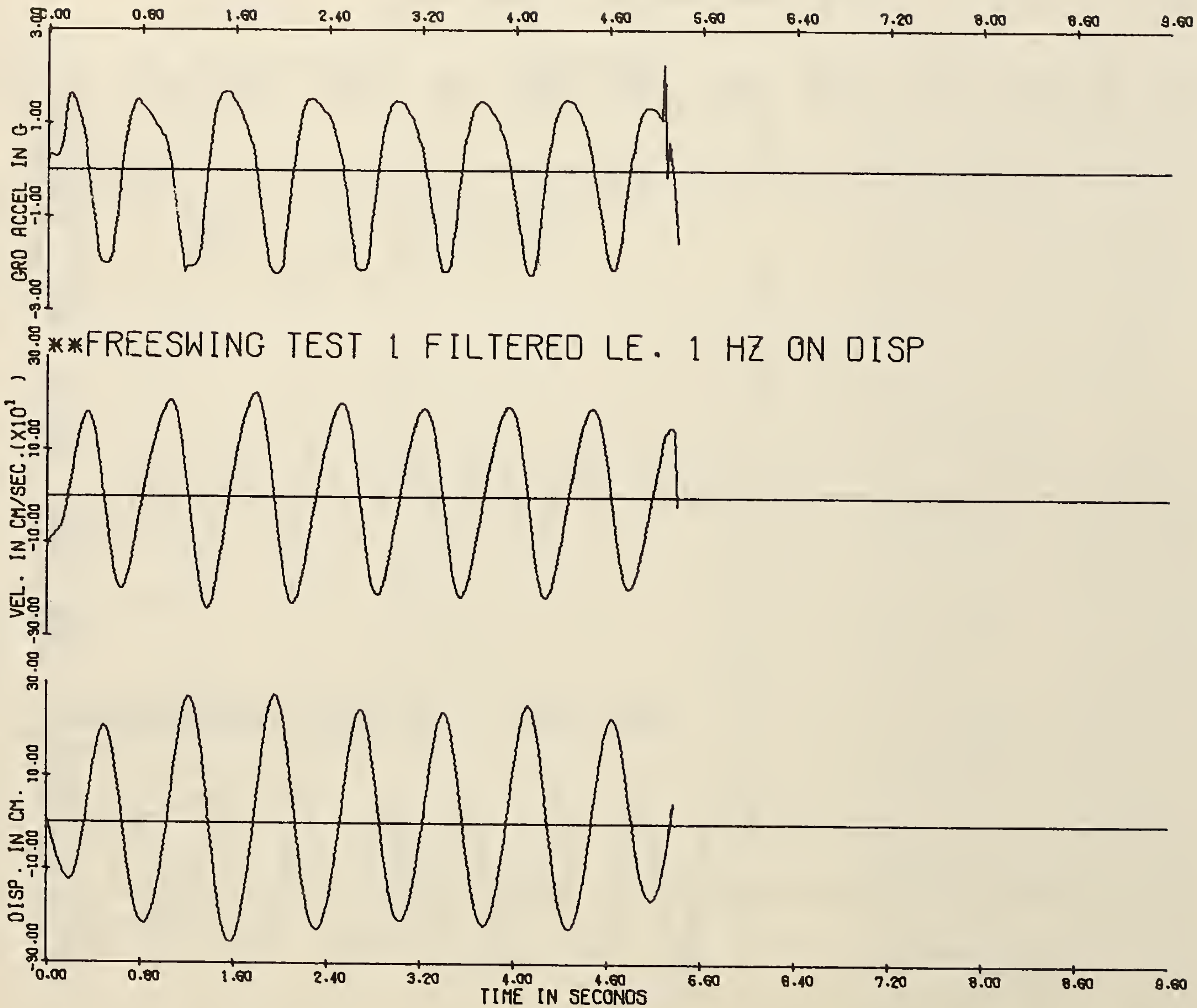


FIGURE 72 PHASE2 RESULTS OF FREE SWING RECORD - N4WAY = 10 VERSION

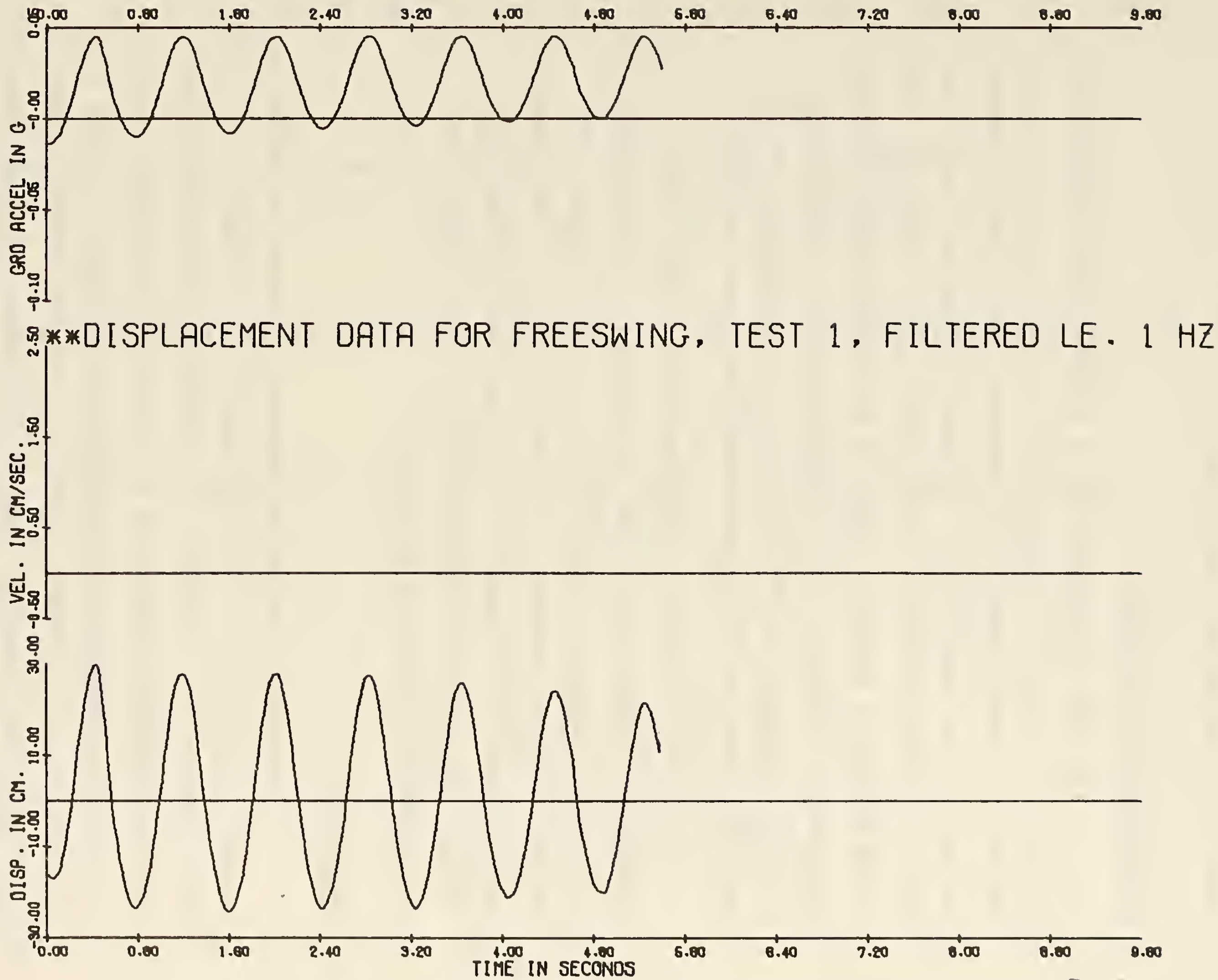


FIGURE 73 FILTERED MEASURED DISPLACEMENT CURVE OF FREE SWING EXPERIMENT - N2WAY = 8 AND N4WAY = 10

of the acceleration record were used in a new integration of the free swing experiment, Figure 74. The results are practically unchanged from Figure 72.

3. Random Motion Experiment

The last experiment was a random motion experiment, Figure 75, in which the pendulum was started from zero and pulled by a rope through displacements simulating a random motion. Since there cannot be negative vertical values when measured from the bottom of the swing (even though the pendulum swings on opposite sides of the bottom position) it is necessary to create a new baseline such that beginning and end displacements are zero. This is accomplished by high-pass filtering the displacement data as shown in Figure 76.

After processing the acceleration input data in PHASE2, the results are shown in Figure 77 for the original Cal. Tech. version; Figures 78, 79, 80, and 81 when the Ormsby filter is changed to 0.1 Hz, 0.3 Hz, 0.5 Hz, and 1 Hz respectively. The change to 1 Hz most nearly matches the integrated time-displacement history to the actual time-displacement history of Figure 76.

4. Discussion

From these limited experiments it would be unwise to declare 1 Hz as the cut-off frequency in the Ormsby filter for strong motion instruments. The piezotron accelerometer used had considerable attenuation in the very low frequencies; and since both experiments (Free Swing and Random) were done with a pendulum, the fundamental cyclic frequency stayed practically constant and affected only the amplitude, not the shape, of the displacement record. These attenuations were corrected

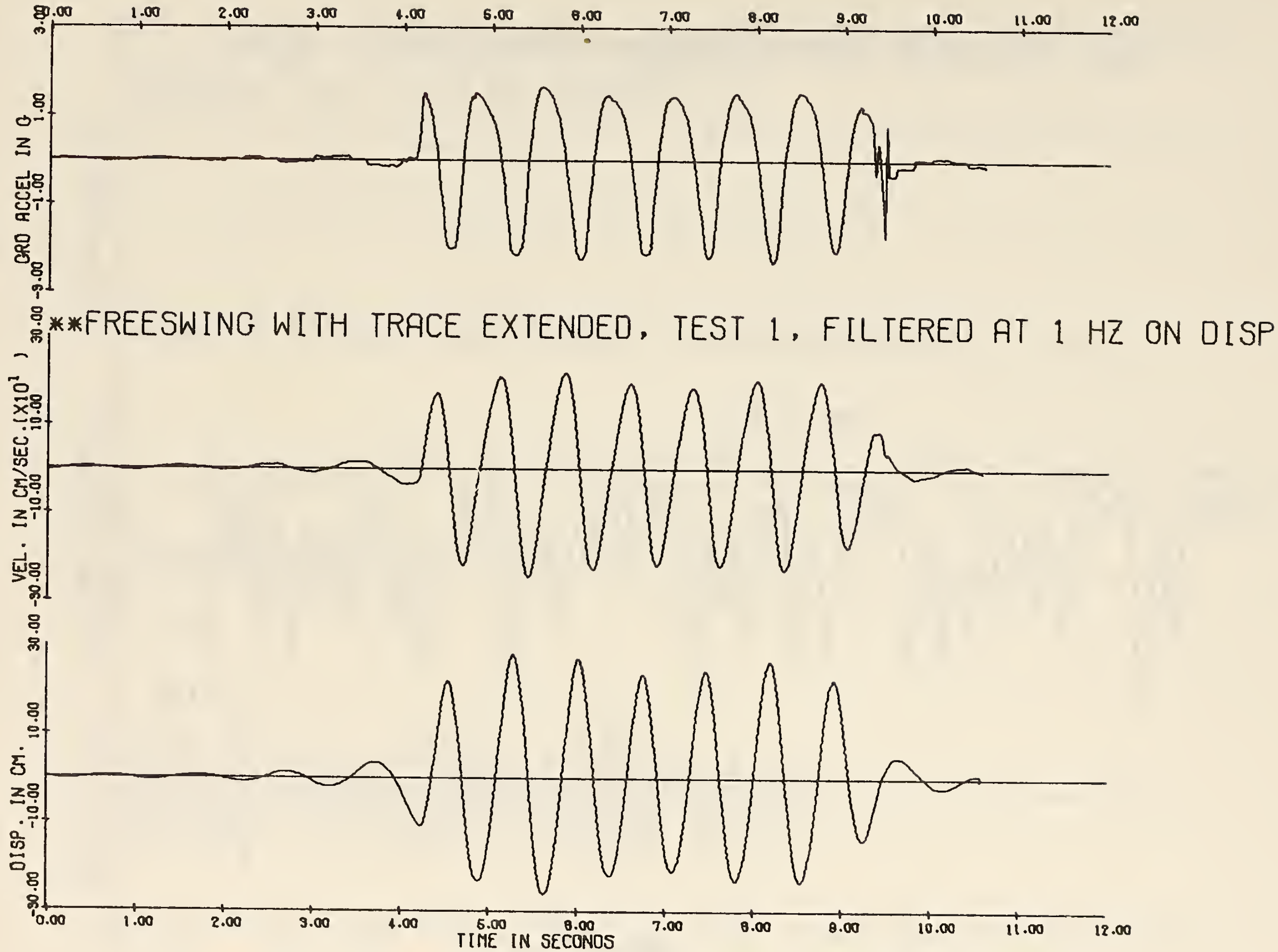


FIGURE 74 PHASE2 RESULTS FOR FREE SWING EXPERIMENT WITH TRACE EXTENDED - N4WAY = 10

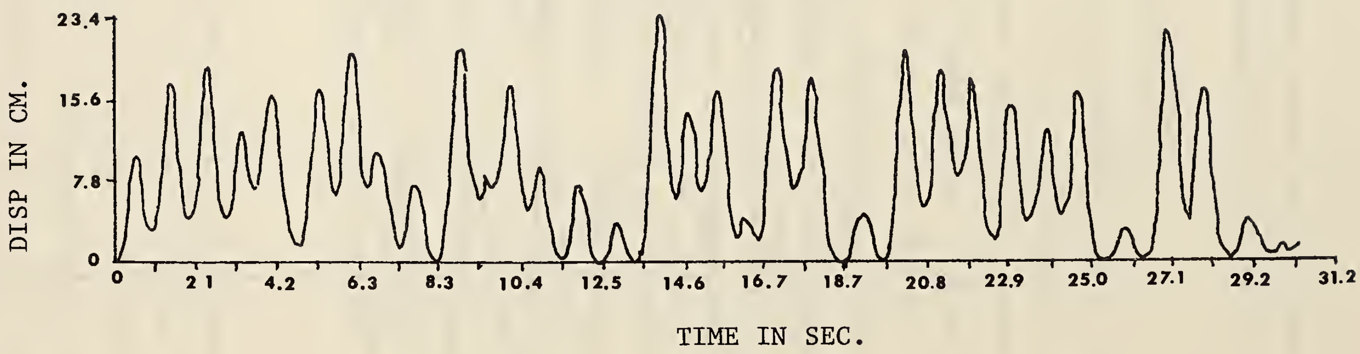


FIGURE 75 RANDOM MOTION EXPERIMENT MEASURED DISPLACEMENT CURVE

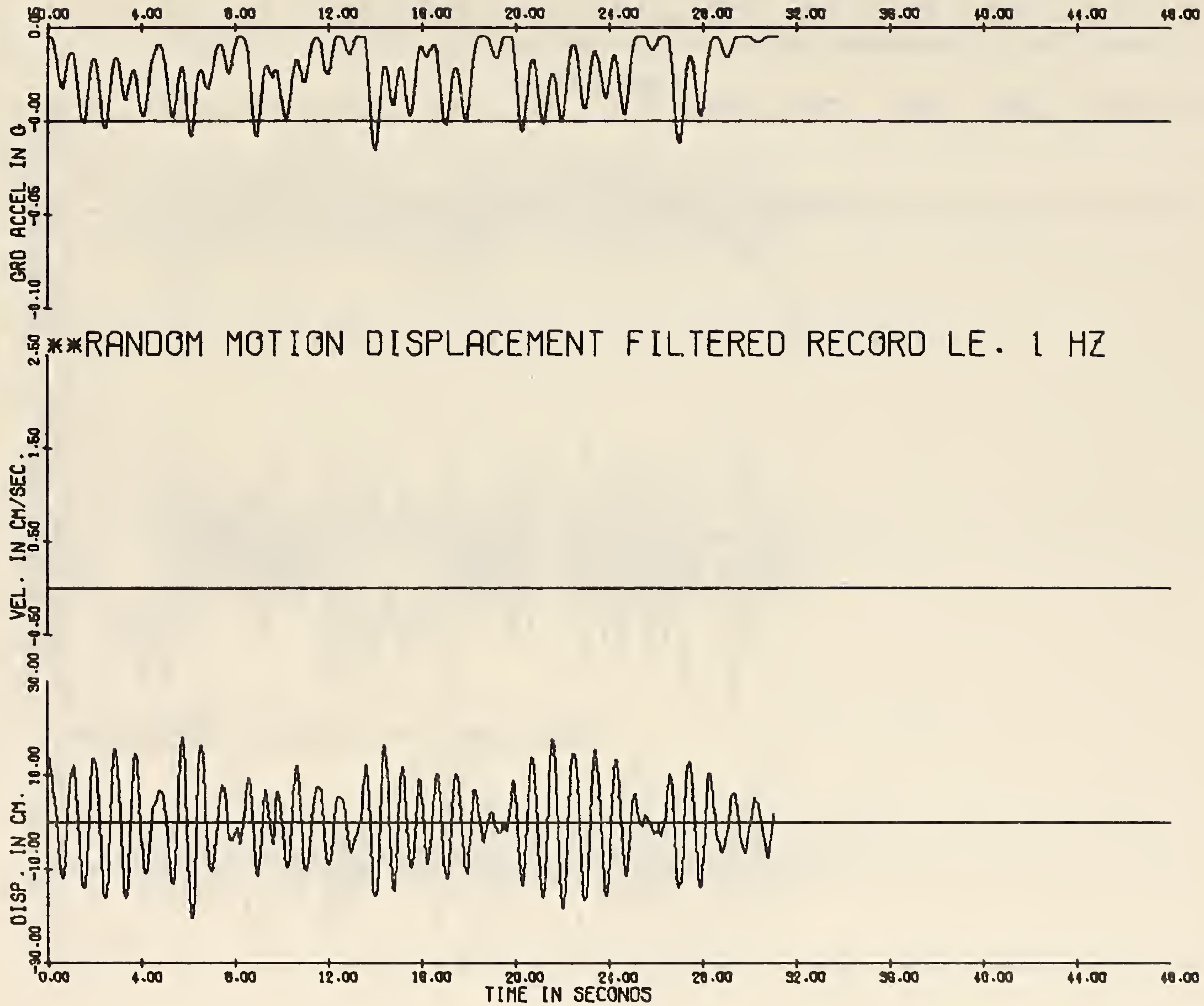


FIGURE 76 FILTERED RANDOM MOTION MEASURED DISPLACEMENT CURVE

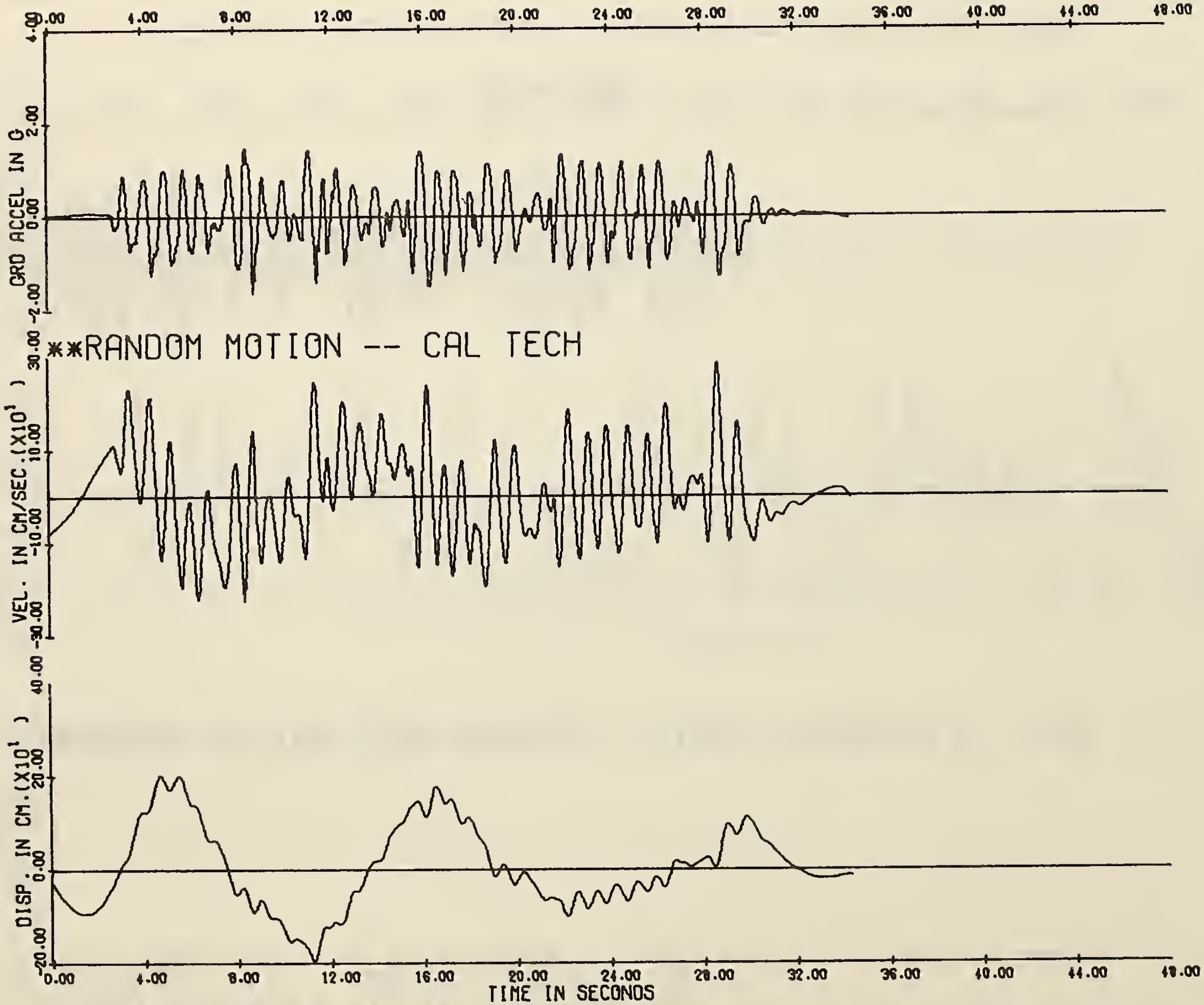


FIGURE 77 PHASE2 RESULTS FOR RANDOM MOTION EXPERIMENT - CAL. TECH. VERSION

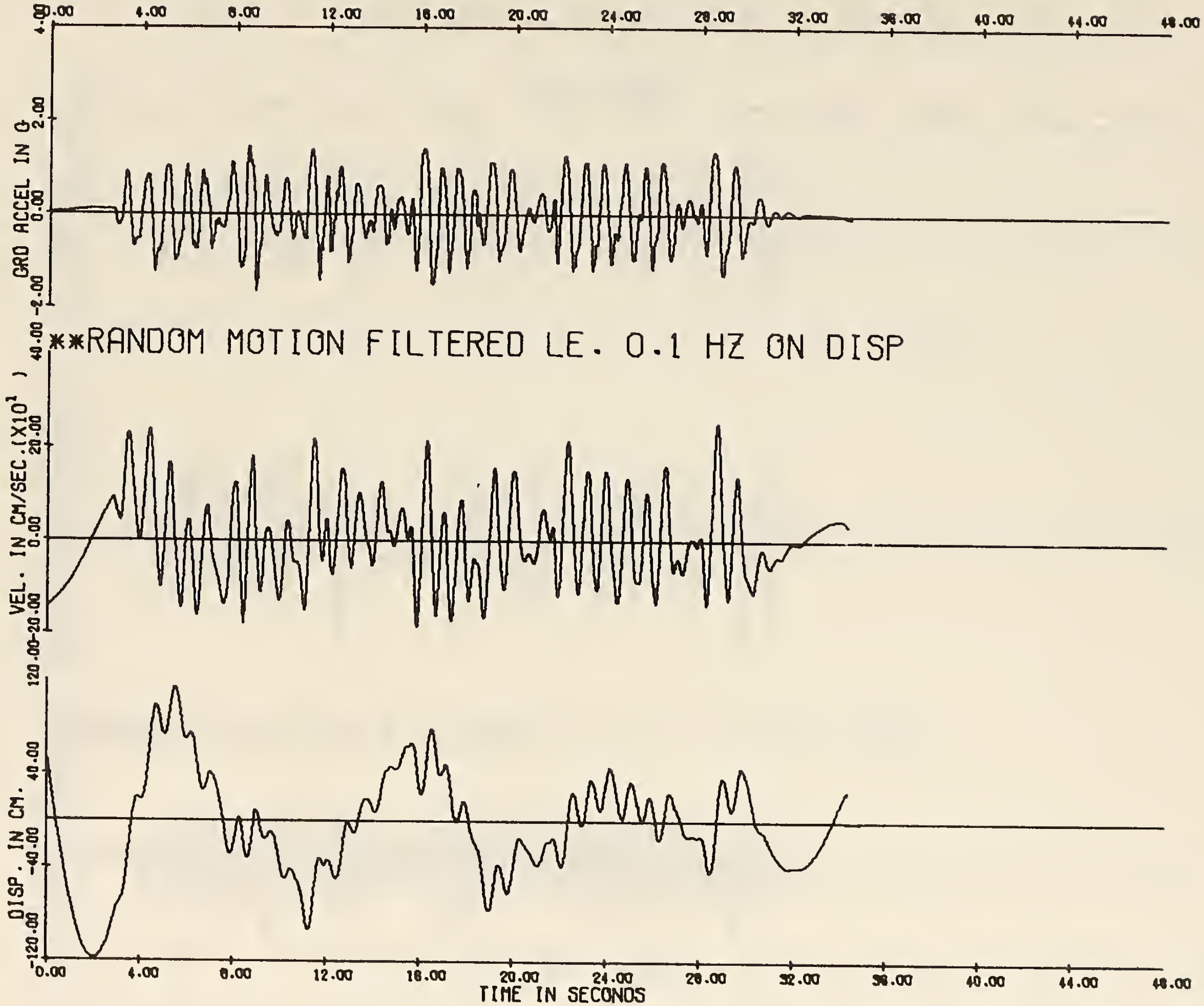


FIGURE 78 PHASE2 RESULTS FOR RANDOM MOTION EXPERIMENT - N4WAY = 7

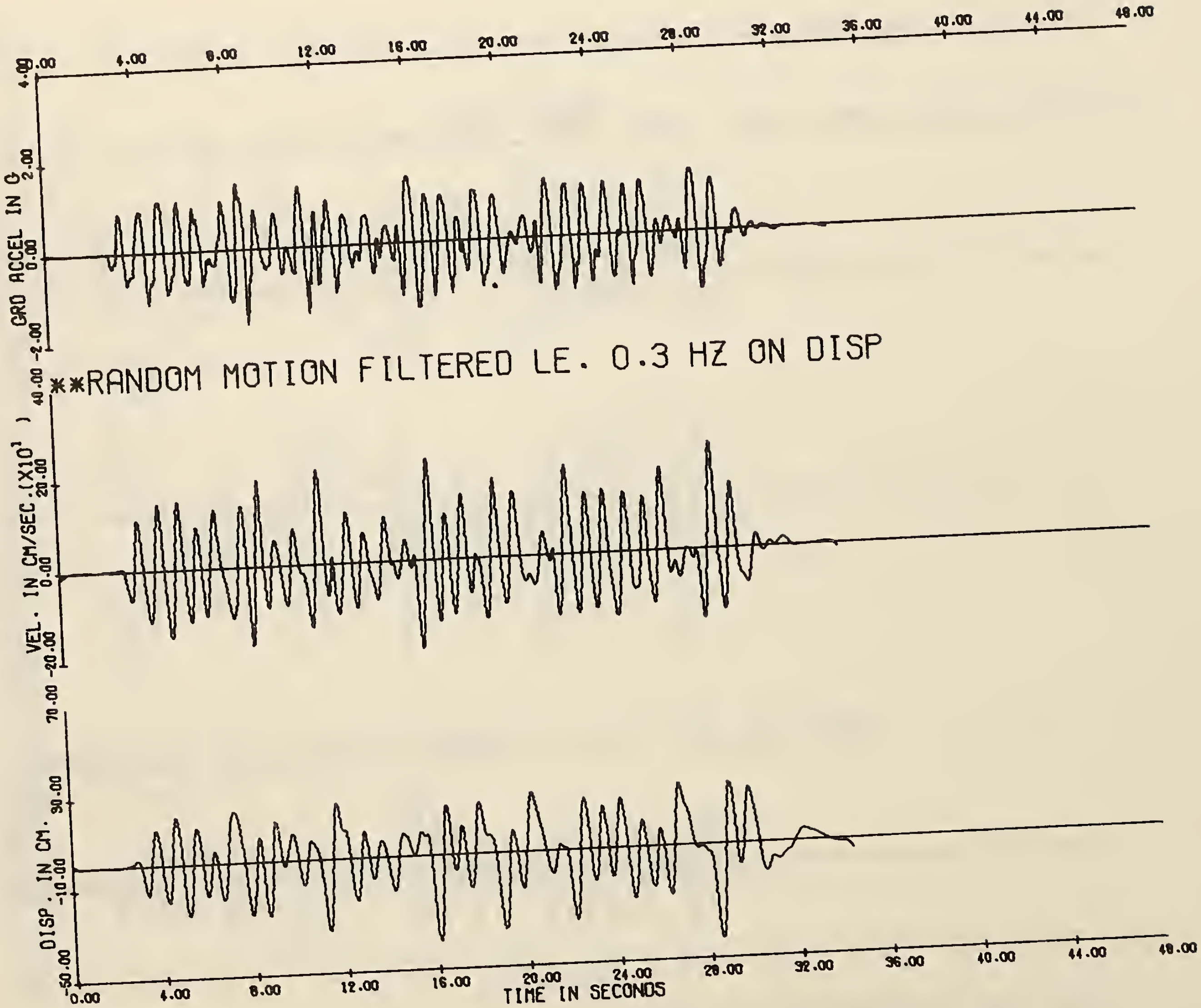


FIGURE 79 PHASE2 RESULTS FOR RANDOM MOTION EXPERIMENT - N4WAY = 8

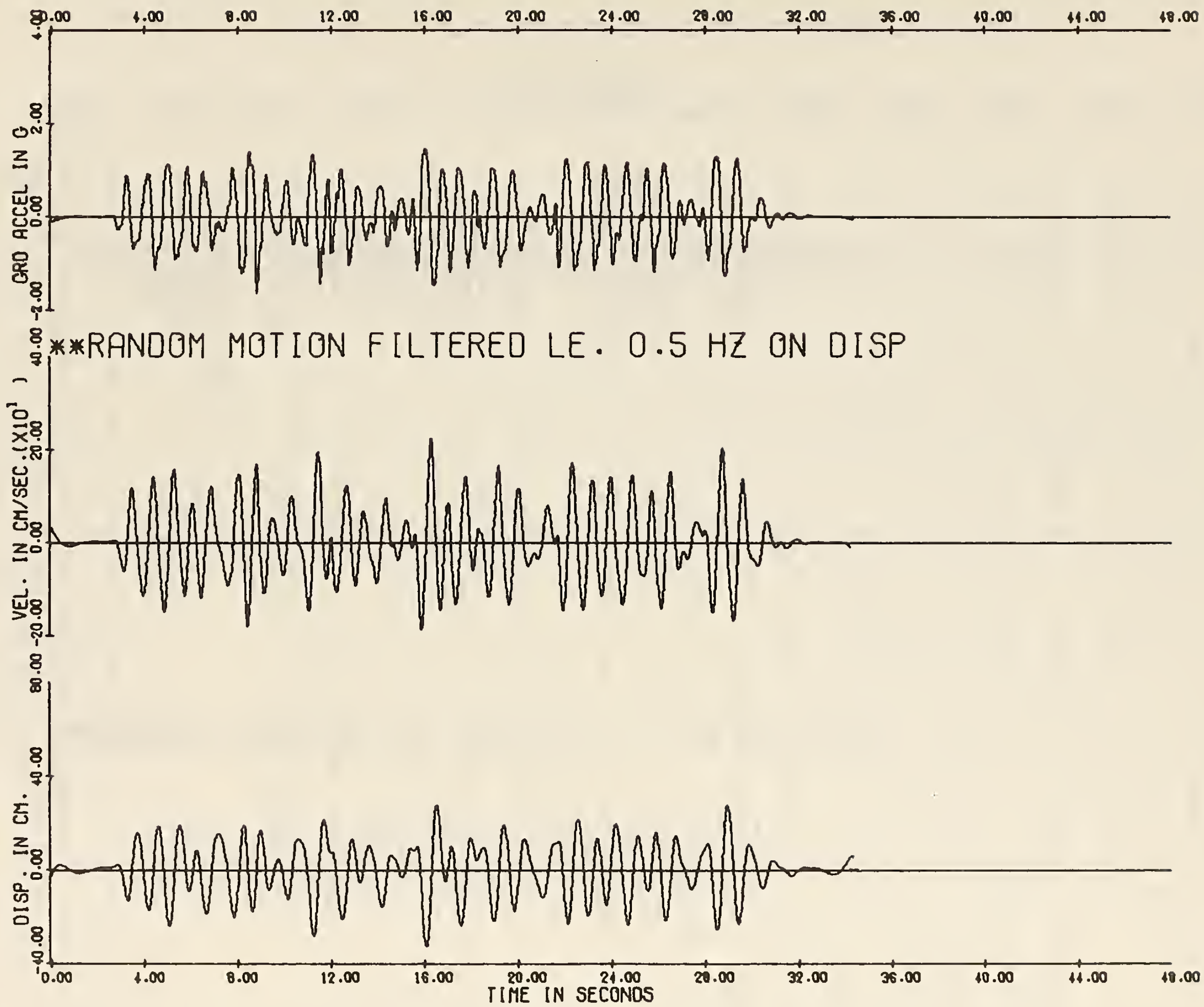


FIGURE 80 PHASE2 RESULTS FOR RANDOM MOTION EXPERIMENT - N4WAY = 5

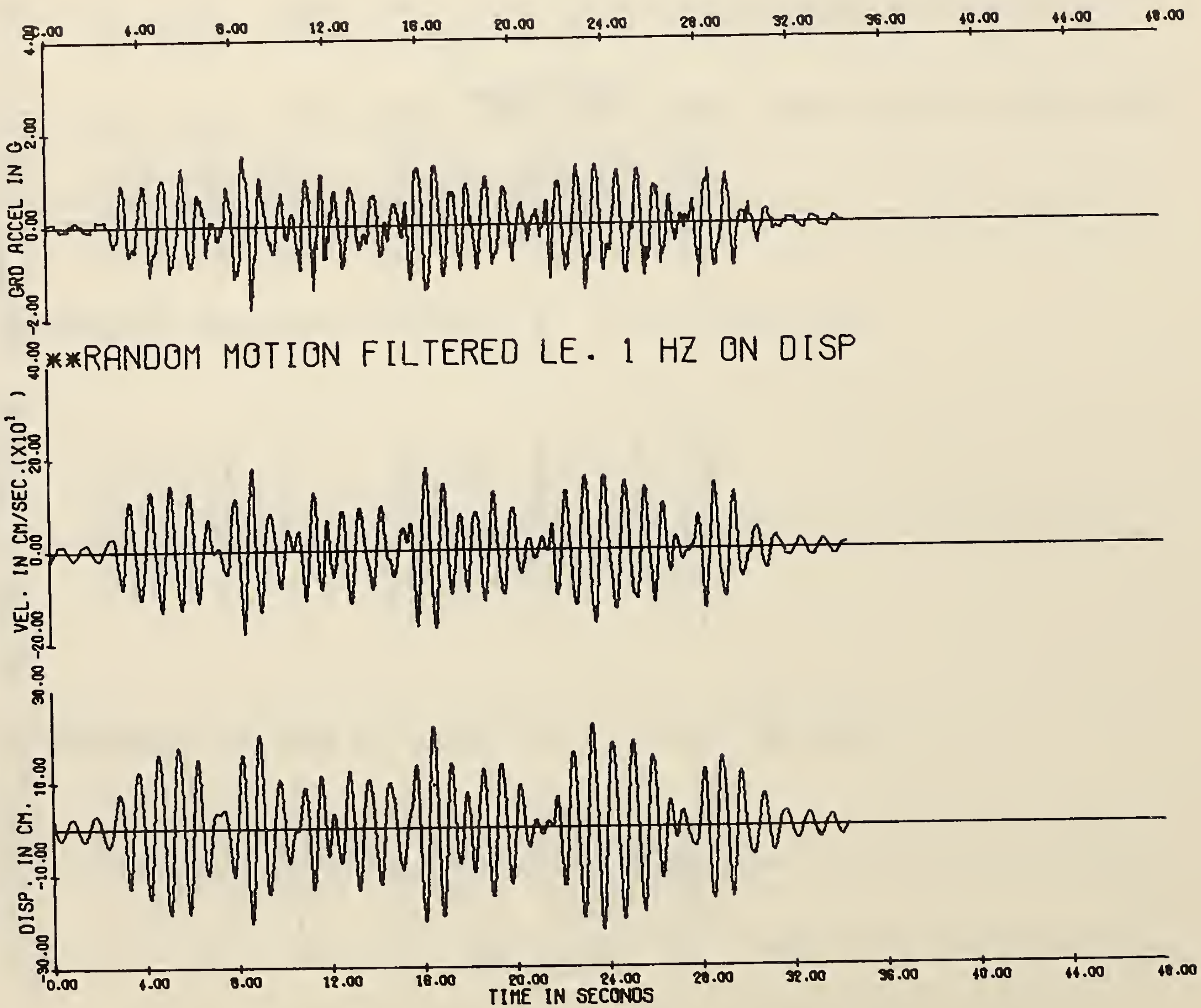


FIGURE 81 PHASE2 RESULTS FOR RANDOM MOTION EXPERIMENT - N4WAY = 10

for in the scaling to match the integrated and measured records. However, a strong motion instrument is capable of a constant sensitivity from 0.1 to 10 Hz, as for example the U. S. C. & G. S. Model II Strong-Motion Seismograph[20], Figure 82. This would probably mean that the Ormsby filter frequency should be set at a lower value, but experiments of the same nature as previously described should be conducted using an actual strong motion instrument (such an instrument was unavailable to the authors), before a lower value can be specified. Further research in this area to determine the upper range of the error frequency band will undoubtedly be necessary for earthquake-type records. One way such a record could be obtained, particularly in the low frequency range, would be to place the strong motion instrument on one end of a waterbed and vibrate the other end by hand. Figure 83 shows some acceleration records that were so obtained in an attempt to simulate earthquake records. However, these were obtained using the Piezotron Accelerometer; what is needed is the same type of record using a strong motion instrument. Coupled with a motion picture camera to record the true acceleration with a background scale, this type of research may prove profitable for future investigators.

5. Displacement Meters

Reference 20 by Trifunac and Lee describes several examples of the use of a displacement meter to check on the accuracy of the Cal. Tech. integration program. Figure 84, taken from reference 20, shows an excellent match in ground displacements obtained from both acceleration and displacement transducers. The displacement meters were long period transducers with T_n = 2 to 11 seconds, recording in the same direction as the acceleration transducers.

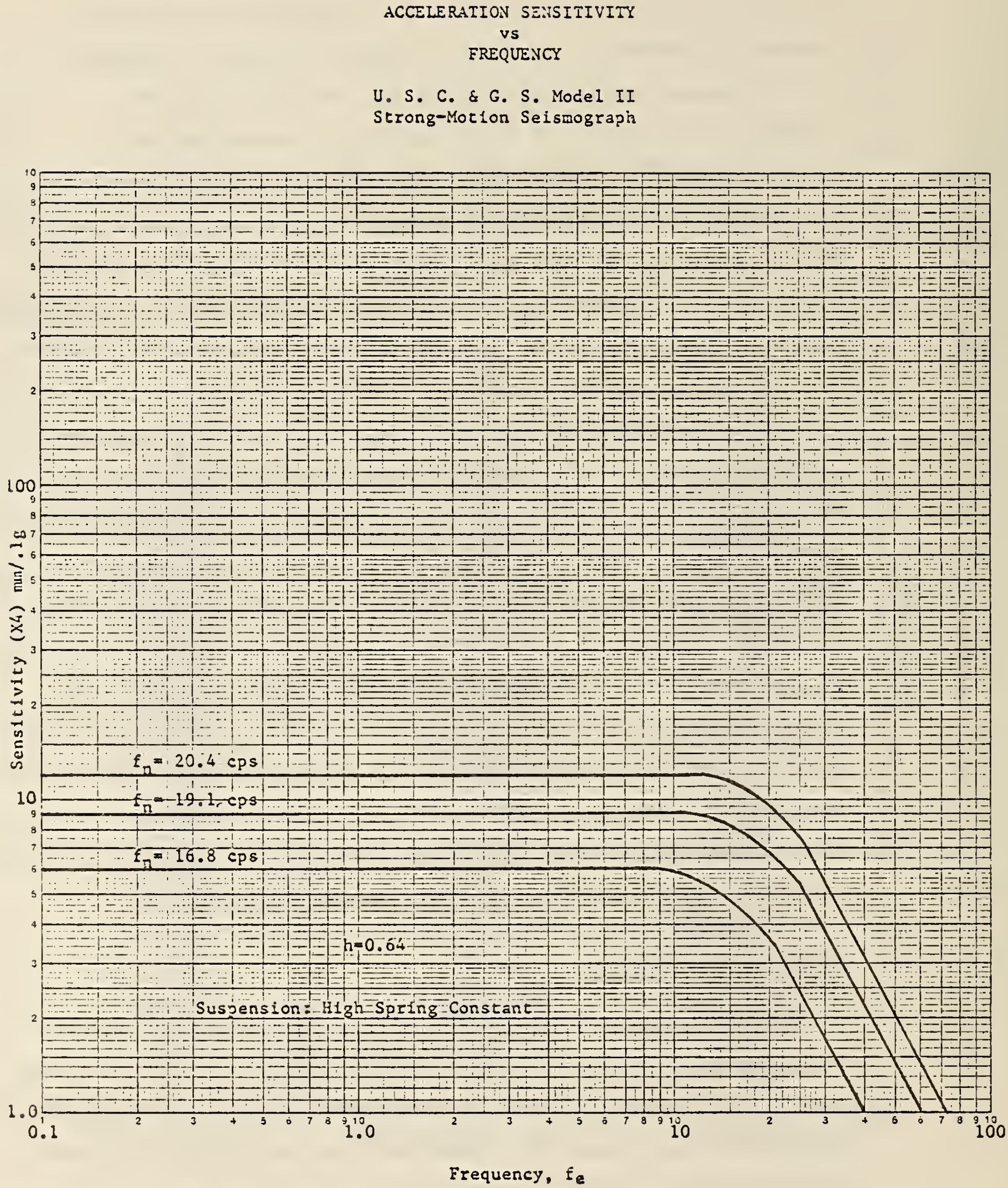


FIGURE 82 FREQUENCY-RESPONSE CURVES FOR U.S.G.S. STRONG MOTION SEISMOGRAPHS

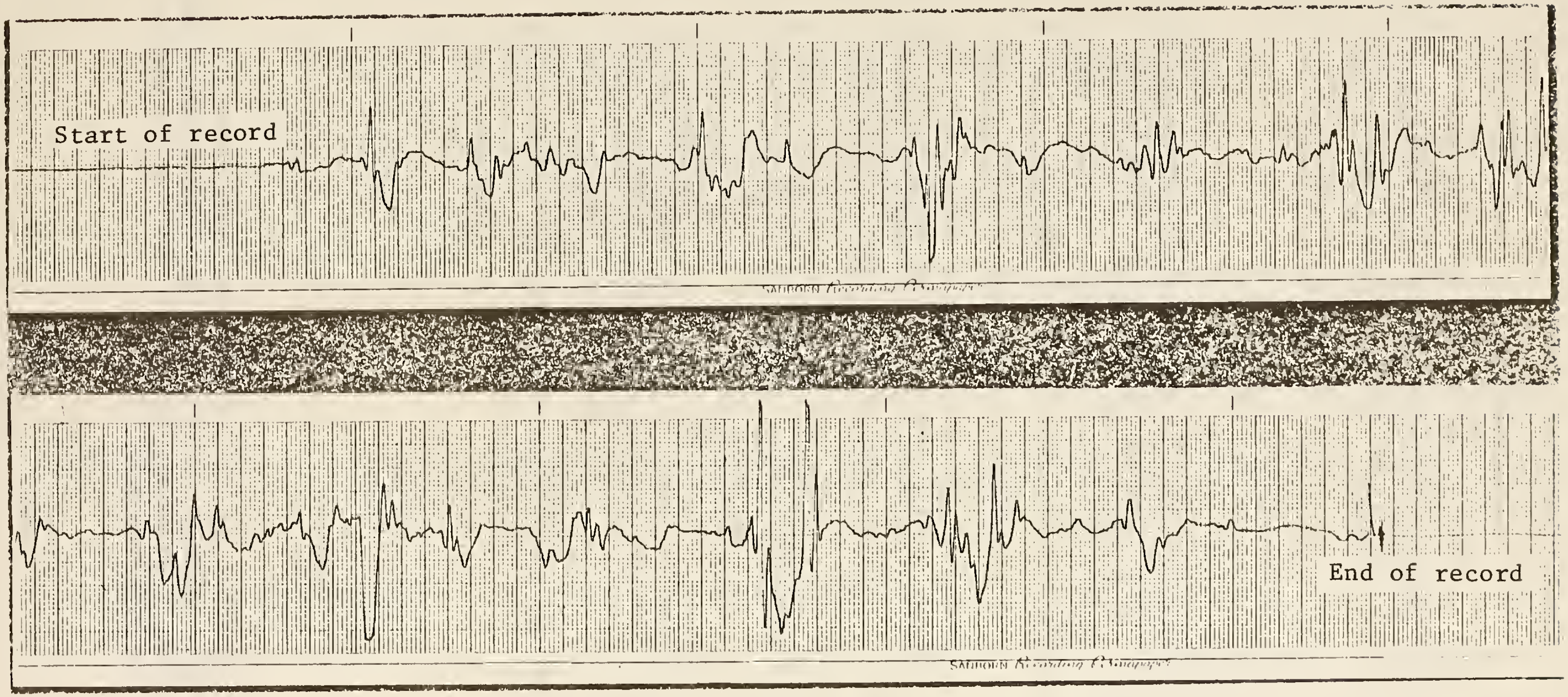


FIGURE 83 RANDOM MOTION ACCELERATION CURVE OBTAINED FROM A WATERBED VIBRATION EXPERIMENT

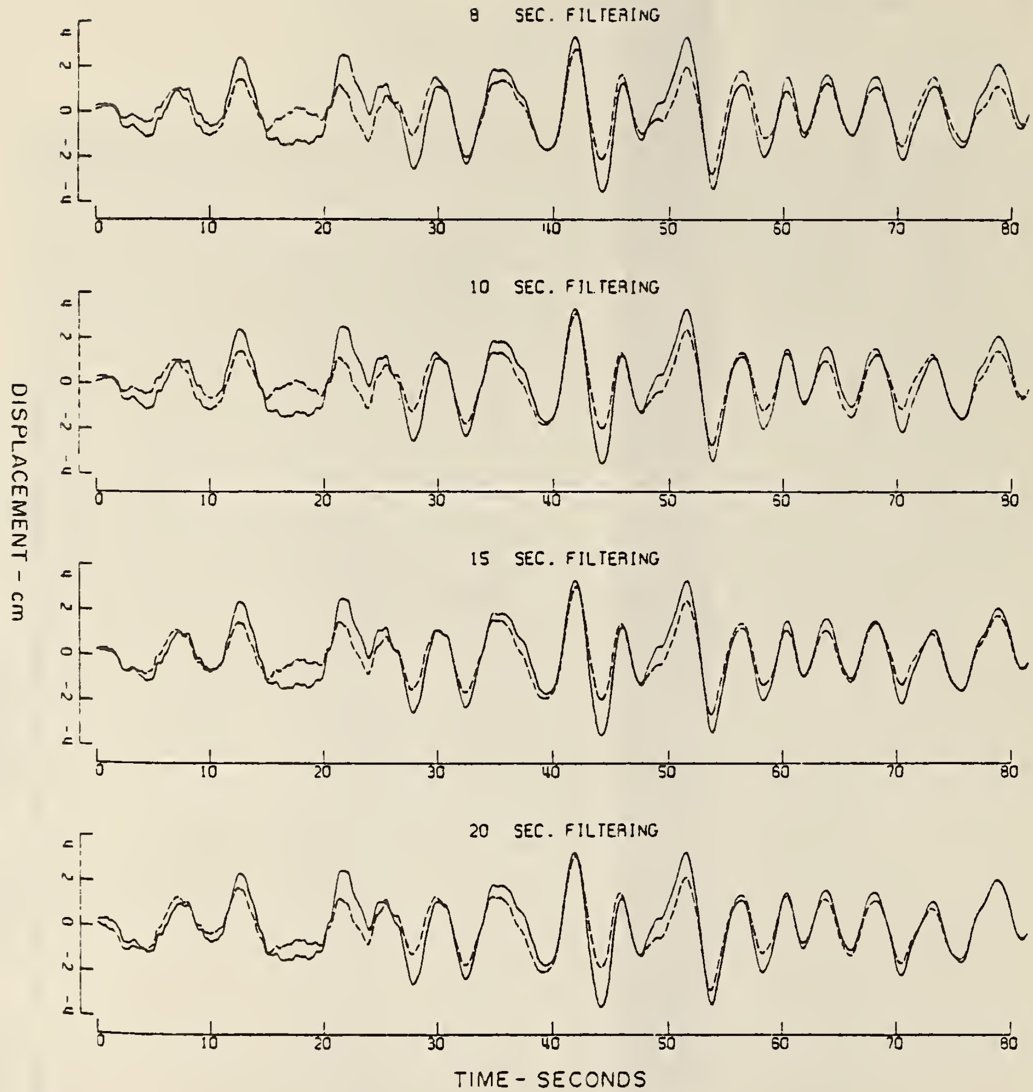


FIGURE 84 GROUND DISPLACEMENTS IN THE SAN FERNANDO EARTHQUAKE AT THE ENGINEERING BUILDING, SANTA ANA, CALIFORNIA (COMP. S04E), --- FROM DISPLACEMENT METER RECORD AND —— FROM ACCELEROGRAPH RECORD (AFTER REFERENCE 20)

To compute the ground displacement from either accelerograms or from displacement records, Trifunac and Lee used the same processing procedure which involved instrument correction, baseline correction, followed by double integration, and the high-pass filtering of velocity and displacement data.

From the previous discussion in this report, it is obvious the same type of record will be obtained if the program methodology itself is manufacturing its own type of displacement record based on error band frequencies. *However, this does not guarantee that the displacements are the true displacements.* To verify this statement the final random motion displacement record was inserted for integration in the Cal. Tech. program with the identical processing as for acceleration data. In order to obtain the same scale of displacement record, because of "integration attenuation", the displacement record was first multiplied by $2.34^2\pi^2 = 54.042$ (the displacement frequency being approximately 1.17 Hz). Figure 85 shows the integration results, which shows the introduction of a long period in the displacement record. Since the input displacement was already filtered at 1 Hz, the long period, although it is not as high an amplitude as for Figure 77, had to be created by the original program methodology.

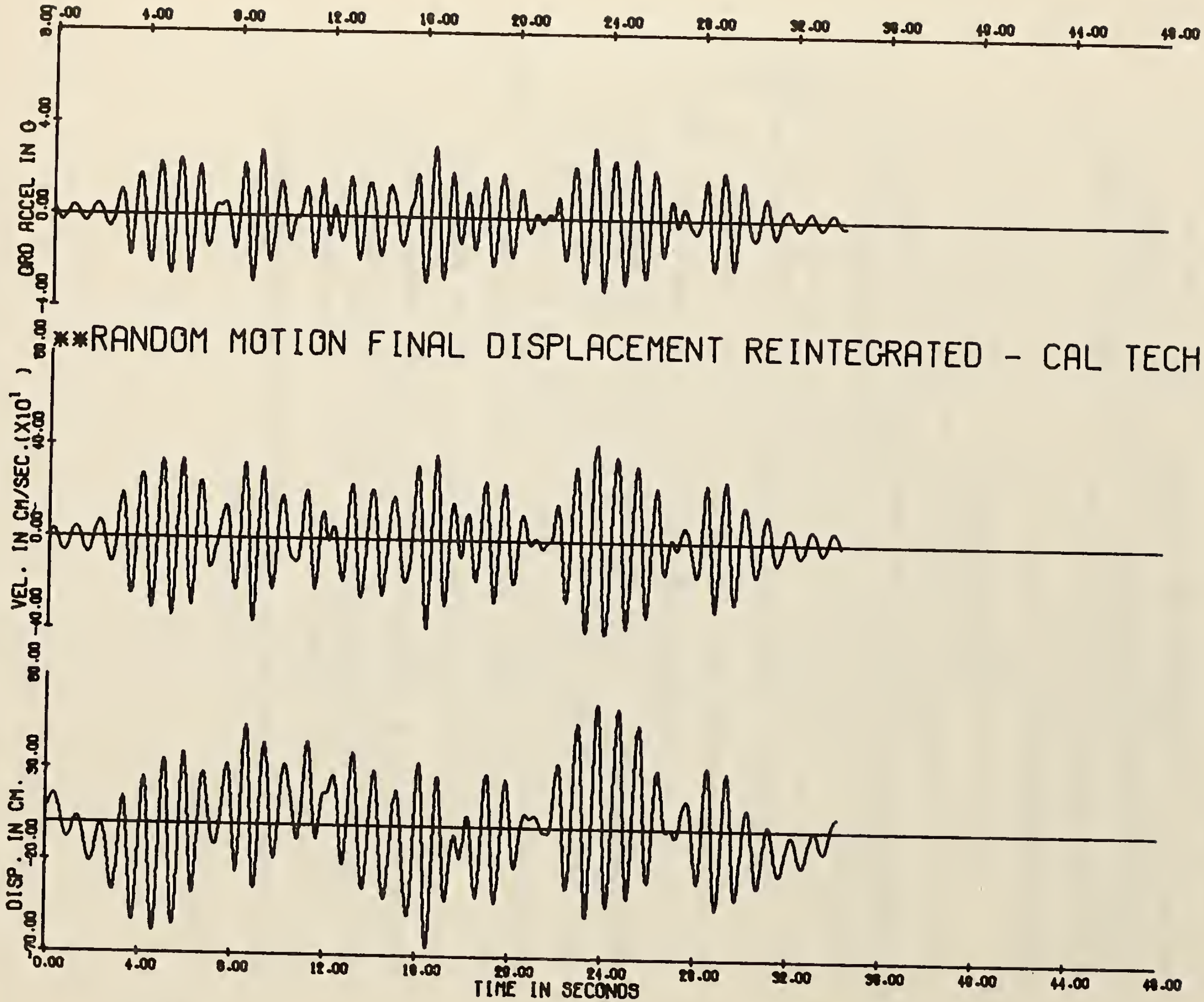


FIGURE 85 PHASE2 RESULTS OBTAINED BY THE INTEGRATION OF THE RANDOM MOTION FINAL DISPLACEMENT CURVE

CONCLUSIONS

1. Figure 86 shows the integrated velocity and displacement of the N76^{o}W Pacoima Dam record in the Cal. Tech. version of PHASE2 via the Electrak digitization, while Figure 87 shows the same integrated record via the Benson-Lehner digitization. In concert with Figure 3 it appears that the Electrak Data-Tablet Digitizer will match reproducible results with the 800 line/inch resolution of Cal. Tech.'s Benson-Lehner machine, while the 200 line/inch resolution Benson-Lehner machine at the Watershed Research Station will not. For efficiency of operation the authors prefer the Electrak machine.

2. Optical scanning is not recommended as a digitizing procedure for strong motion records. The procedure is too costly and when peaks and valleys in the record are at close spacing, the accuracy is much reduced.

3. Digitizing by hand using a Microfische Film Reader with a contact printing screen bacground grid is an accurate method of digitization when a commercial machine is unavailable. Although this method is tedious and time consuming, the accuracy of this method of digitization can match that of the Electrak machine. A free swing record was redigitized via the microfische method and integrated giving identical results to Figure 74.

4. The Cal. Tech. program will give true time-displacement histories under certain conditions. These are:

    A. The true velocity at the beginning and end of the record is zero.

    B. The true displacement at the beginning and end of the record is zero.

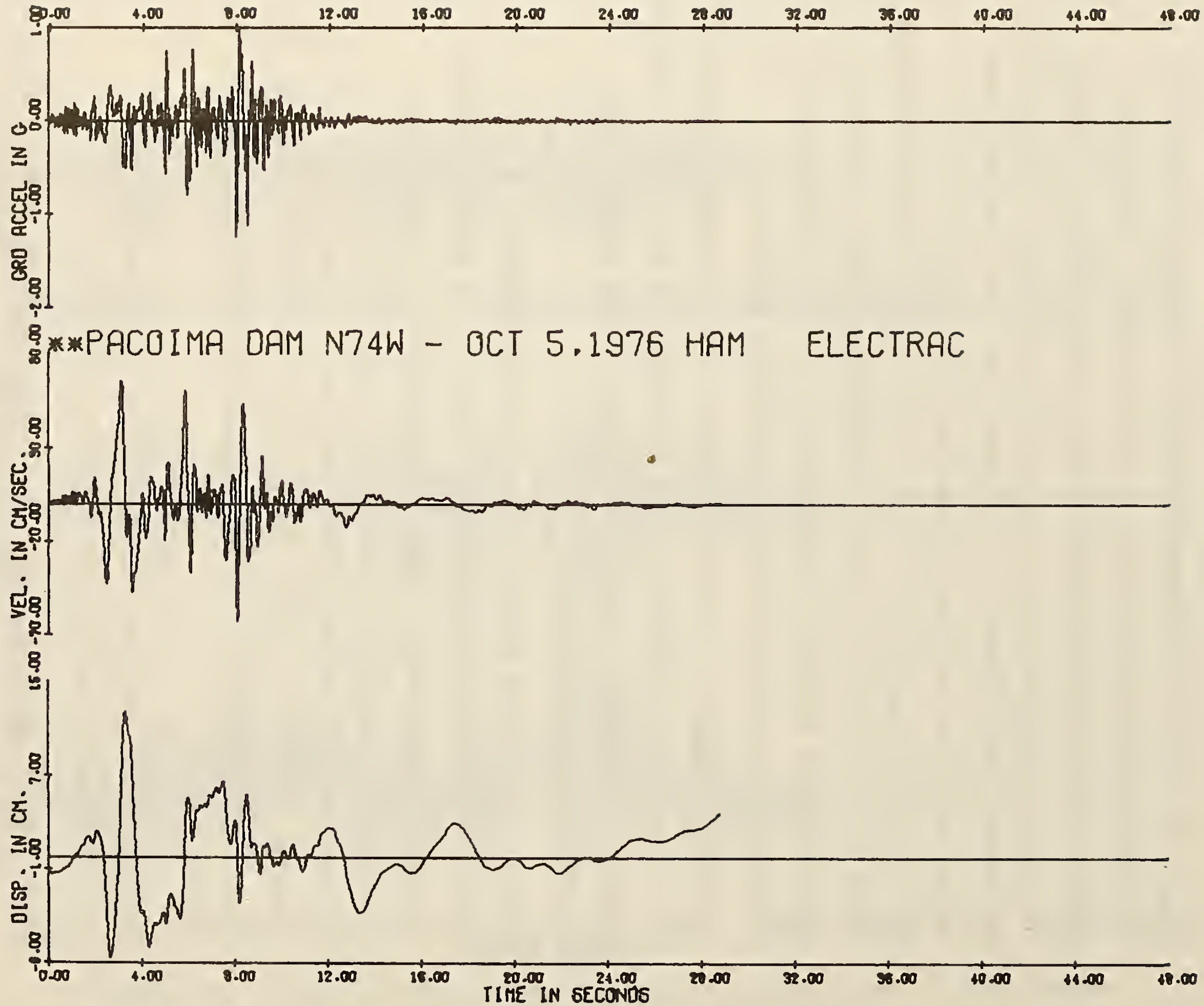


FIGURE 86 PHASE2 RESULTS FOR N76°W PACOIMA DAM RECORD - CAL. TECH. VERSION. DIGITIZATION VIA ELECTRAK MACHINE

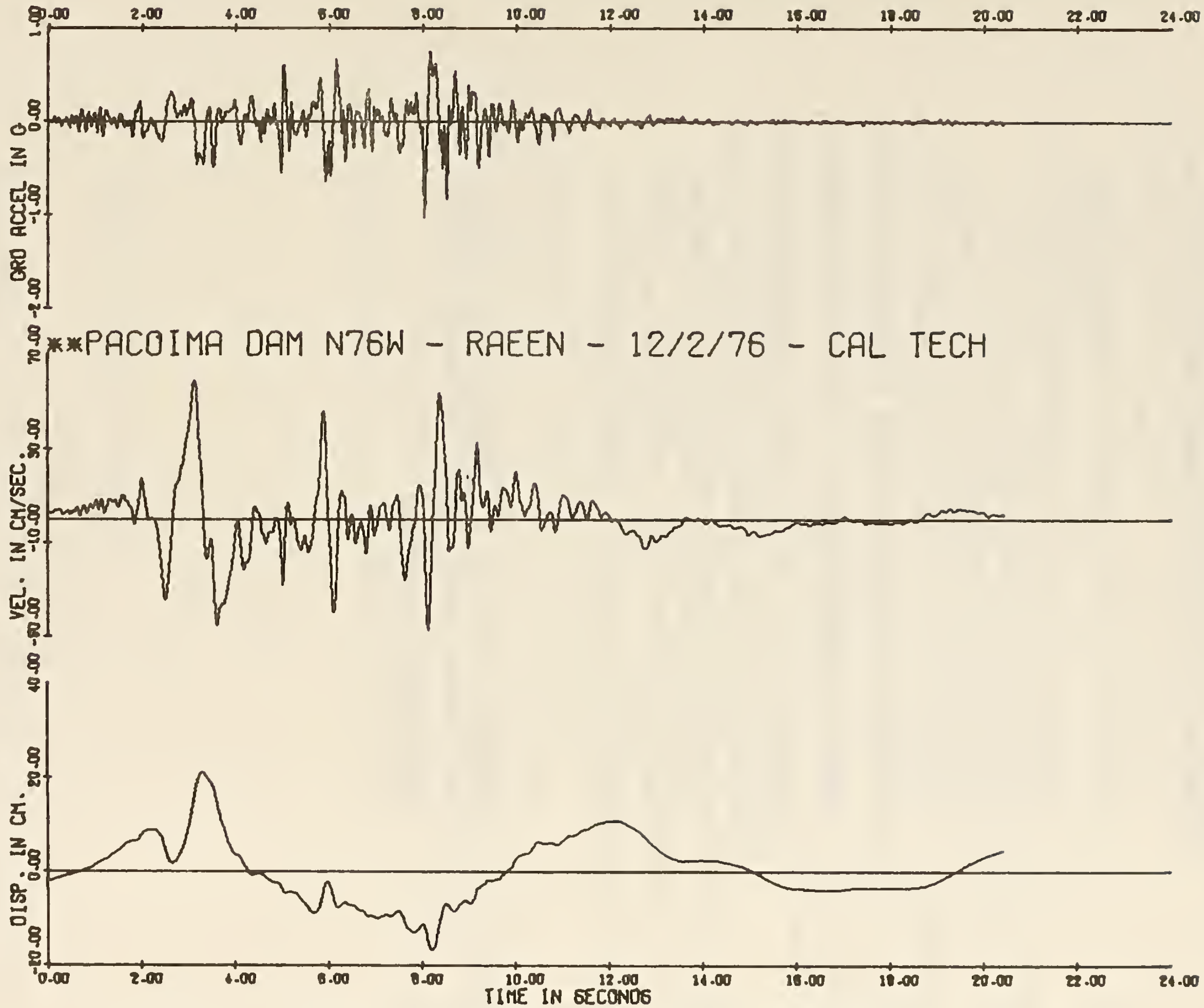


FIGURE 87 PHASE2 RESULTS FOR N76°W PACOIMA DAM RECORD - CAL. TECH. VERSION. DIGITIZATION VIA BENSON-LEHNER MACHINE

C. The Ormsby filter frequency is increased from $\frac{1}{16}$ Hz to a value above the highest error frequency, but below the lowest frequency of interest in the record. If these two frequencies are too close to each other, the program cannot guarantee a true time-displacement history. The integration methodology using the process of leastsquaring, Holoway, and Ormsby filtering produces errors. From the experiments conducted in this investigation using a Piezotron Accelerometer it appears that the error frequency may be extended to 0.5 Hz. Further investigations may be necessary with strong motion instruments and simulated earthquake displacements measured independently from the earthquake source to refine the change in Ormsby frequency.

D. Verification of the accuracy of the integration program for an accelerogram cannot be done with a displacement meter record processed through the same computer program.

APPENDIX A

Least Squares Theory for Integration Program

1.1 Introduction

One common assumption in digitizing any record is that a linear variation exists between the digitized points. If the record is an acceleration record, then the integrated velocity between points varies as a parabola, and the integrated displacement between points varies as a cubic equation. Corrections to the acceleration record are usually made as a polynomial of varying degree with constant coefficients -- the most common being the straight line or parabolic corrections. The constants are evaluated by a minimization procedure, so that the root mean square value of the corrected velocity is a minimum.

1.2 Parabolic Line Correction - Boyce's Program, New Zealand

To an acceleration term corresponding to a time t_i a parabolic line correction takes the form of:

$$C_0 + C_1 t_i + C_2 t_i^2$$

Let $a = C_0$ $\quad 2b = C_1$ $\quad 3c = C_2$

Then the correction becomes:

$$a + 2bt_i + 3ct_i^2$$

The values of a, b and c will be chosen such that the root mean square value of the corrected velocity is a minimum. Writing uncorrected acceleration and velocity terms as A_u and V_u respectively and corrected terms as A_c and V_c we have:

$$V_c = V_0 + \int_0^T A_u dt + at + bt^2 + ct^3$$

$$= V_0 + V_u + at + bt^2 + ct^3$$

We wish to minimize $\int_0^T V_c^2 dt$ where T is the total record time in seconds; therefore, we must solve the following simultaneous equations:

$$\int_\delta^T V_c \frac{\delta V_c}{\delta V_0} dt = 0 \qquad \text{where} \quad \frac{\delta V_c}{\delta V_0} = 1$$

$$\int_0^T V_c \frac{\delta V_c}{\delta a} dt = 0 \qquad \text{where} \quad \frac{\delta V_c}{\delta a} = t$$

$$\int_0^T V_c \frac{\delta V_c}{\delta b} dt = 0 \qquad \text{where} \quad \frac{\delta V_c}{\delta b} = t^2$$

$$\int_0^T V_c \frac{\delta V_c}{\delta c} dt = 0 \qquad \text{where} \quad \frac{\delta V_c}{\delta c} = t^3$$

These four equations give:

$$\begin{aligned}
V_0 T + \frac{aT^2}{2} + \frac{bT^3}{3} + \frac{cT^4}{4} &= -\int V_u dt = W \\
V_0 \frac{T^2}{2} + \frac{aT^3}{3} + \frac{bT^4}{4} + \frac{cT^5}{5} &= -\int V_u t dt = X \\
V_0 \frac{T^3}{3} + \frac{aT^4}{4} + \frac{bT^5}{5} + \frac{cT^6}{6} &= -\int V_u t^2 dt = Y \\
V_0 \frac{T^4}{4} + \frac{aT^5}{5} + \frac{bT^6}{6} + \frac{cT^7}{7} &= -\int V_u t^3 dt = Z \qquad \text{(A.1)}
\end{aligned}$$

If the initial velocity is made equal to zero then the first row and column is eliminated so that only the following three equations are solved for a, b, and c by Cramer's rule:

$$\frac{aT^3}{3} + \frac{bT^4}{4} + \frac{cT^5}{5} = -\int V_u t\,dt = X$$

$$\frac{aT^4}{4} + \frac{bT^5}{5} + \frac{cT^6}{6} = -\int V_u t^2 dt = Y$$

$$\frac{aT^5}{5} + \frac{bT^6}{6} + \frac{cT^7}{7} = -\int V_u t^3 dt = Z \qquad \text{(A.2)}$$

Denominator expansion -- expanded by top row:

$$\begin{array}{c} \begin{array}{ccc} a & b & c \end{array} \\ \begin{vmatrix} \frac{T^3}{3} & \frac{T^4}{4} & \frac{T^5}{5} \\ \frac{T^4}{4} & \frac{T^5}{5} & \frac{T^6}{6} \\ \frac{T^5}{5} & \frac{T^6}{6} & \frac{T^7}{7} \end{vmatrix} \end{array}$$

$$= \frac{T^3}{3}\ [T^{12}](\frac{1}{35} - \frac{1}{36}) - \frac{T^4}{4}\ [T^{11}](\frac{1}{28} - \frac{1}{30}) + \frac{T^5}{5}\ [T^{10}](\frac{1}{24} - \frac{1}{25})$$

$$= T^{15}[.0002645503 - .0005952381 + .0003333333]$$

$$= .0000026455T^{15}$$

Numerator expansion for a -- expanded by left column:

$$\begin{vmatrix} X & \frac{T^4}{4} & \frac{T^5}{5} \\ Y & \frac{T^5}{5} & \frac{T^6}{6} \\ Z & \frac{T^6}{6} & \frac{T^7}{7} \end{vmatrix}$$

$$= X[T^{12}](\frac{1}{35} - \frac{1}{36}) - Y[T^{11}](\frac{1}{28} - \frac{1}{30}) + Z[T^{10}](\frac{1}{24} - \frac{1}{25})$$

$$= .0007936508T^{12}X - .0023809524T^{11}Y + .0016666667T^{10}Z$$

Dividing through by the denominator and substituting for X, Y, and Z we get:

$$C_0 = a = -300\int_0^T \frac{V_u(t)tdt}{T^3} + 900\int_0^T \frac{V_u(t)t^2dt}{T^4}$$

$$- 630\int_0^T \frac{V_u(t)t^3dt}{T^5} \tag{A.3}$$

Numerator expansion for b -- expanded by middle column:

$$\begin{vmatrix} \frac{T^3}{3} & X & \frac{T^5}{5} \\ \frac{T^4}{4} & Y & \frac{T^6}{6} \\ \frac{T^5}{5} & Z & \frac{T^7}{7} \end{vmatrix}$$

$$= -X[T^{11}](\frac{1}{28} - \frac{1}{30}) + Y[T^{10}](\frac{1}{21} - \frac{1}{25}) - Z[T^9](\frac{1}{18} - \frac{1}{20})$$

$$= - .002308924T^{11}X + .0076190476T^{10}Y - .0055555556T^9Z$$

Dividing through by the denominator and substituting for X, Y, and Z we get:

$$b = 900\int_0^T \frac{V_u(t)tdt}{T^4} -2880\int_0^T \frac{V_u(t)t^2dt}{T^5} - 2100\int_0^T \frac{V_u(t)t^3dt}{T^6}$$

or

$$C_1 = 2b = 1800\int_0^T \frac{V_u(t)tdt}{T^4} - 5760\int_0^T \frac{V_u(t)t^2dt}{T^5}$$

$$+ 4200\int_0^T \frac{V_u(t)t^3dt}{T^6} \tag{A.4}$$

Numerator expansion for c -- expanded by right column:

$$\begin{vmatrix} \frac{T^3}{3} & \frac{T^4}{4} & X \\ \frac{T^4}{4} & \frac{T^5}{5} & Y \\ \frac{T^5}{5} & \frac{T^6}{6} & Z \end{vmatrix}$$

$$= X[T^{10}](\frac{1}{24} - \frac{1}{25}) - Y[T^9](\frac{1}{18} - \frac{1}{20}) + Z[T^8](\frac{1}{15} - \frac{1}{16})$$

$$= .0016666667T^{10}X - .005555556T^9Y + .0041666667T^8Z$$

Dividing through by the denominator and substituting for X, Y, and Z we get:

$$c = -630\int_0^T \frac{V_u(t)tdt}{T^5} - 2100\int_0^T \frac{V_u(t)t^2dt}{T^6} - 1575\int_0^T \frac{V_u(t)t^3dt}{T^7}$$

or

$$C_2 = 3c = -1890\int_0^T \frac{V_u(t)tdt}{T^5} + 6300\int_0^T \frac{V_u(t)t^2dt}{T^6}$$

$$- 4725\int_0^T \frac{V_u(t)t^3dt}{T^7} \qquad \text{(A.5)}$$

Equations (A.3), (A.4), and (A.5) are used in Boyce's Program for the leastsquare procedure when the initial velocity is considered to be zero. When $V_0 \neq 0$, then the evaluation of equation (A.1) results in the following solution:

$$V_0 = 16\int_0^T \frac{V_u dt}{T} - 120\int_0^T \frac{V_u t dt}{T^2} + 240\int_0^T \frac{V_u t^2 dt}{T^3} + 140\int_0^T \frac{V_u t^3 dt}{T^4} \qquad (A.6)$$

$$C_0 = -120\int_0^T \frac{V_u dt}{T^2} + 1200\int_0^T \frac{V_u t dt}{T^3} - 2700\int_0^T \frac{V_u t^2 dt}{T^4} + 1680\int_0^T \frac{V_u t^3 dt}{T^5} \qquad (A.7)$$

$$C_1 = 480\int_0^T \frac{V_u dt}{T^3} - 5400\int_0^T \frac{V_u t dt}{T^4} + 12960\int_0^T \frac{V_u t^2 dt}{T^5} - 8400\int_0^T \frac{V_u t^3 dt}{T^6} \qquad (A.8)$$

$$C_2 = -420\int_0^T \frac{V_u dt}{T^4} + 5040\int_0^T \frac{V_u t dt}{T^5} - 12600\int_0^T \frac{V_u t^2 dt}{T^6} + 8400\int_0^T \frac{V_u t^3 dt}{T^7} \qquad (A.9)$$

Equations (A.6) through (A.9) are used in Boyce's Program.

1.3 Evaluation of Integrals - Boyce's Program

Simpson's Rule [21] is a well-known quadratic formula for the evaluation of an area under a curve by numerical analysis, provided the abscissa of the curve is divided into an even number of lengths (Figure A-1), denoted by h.

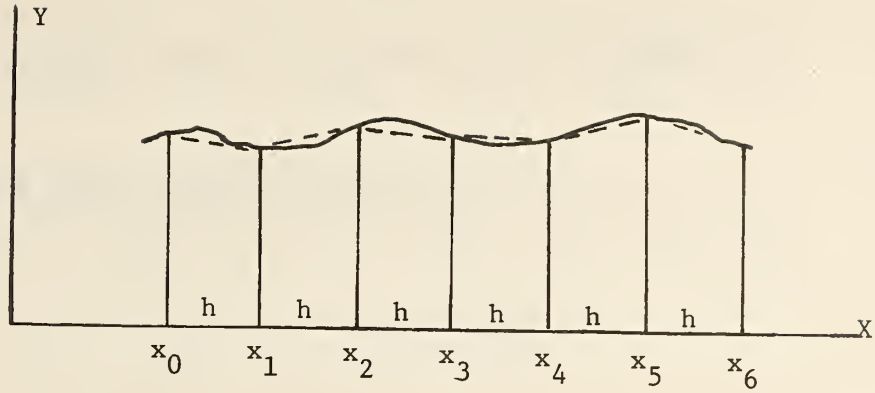


FIGURE A-1 INTERVALS FOR SIMPSON'S RULE

The area is evaluated by considering pairs of intervals:

$$\int_{x_0}^{x_2} f(x)dx = \frac{h}{3}\left[f(x_0) + 4f(x_1) + f(x_2)\right]$$

$$\int_{x_2}^{x_4} f(x)dx = \frac{h}{3}\left[f(x_2) + 4f(x_3) + f(x_4)\right]$$

$$\int_{x_4}^{x_6} f(x)dx = \frac{h}{3}\left[f(x_4) + 4f(x_5) + f(x_6)\right]$$

etc.

For unequally spaced digitized time data, it is necessary to divide each time increment into two equal divisions and consider h = time(i + 1) - time(i). The least square integral equations then become:

$$\int_{0}^{T} V_u t dt = \sum \frac{t_{i+1} - t_i}{2(3)} \left[V_i t_i + 4V_{i+\frac{1}{2}} \left(\frac{t_i + t_{i+1}}{2}\right) + V_{i+1} t_{i+1}\right]$$

$$= \sum \frac{t_{i+1} - t_i}{6} [V_i t_i + 2V_{i+\frac{1}{2}} (t_i + t_{i+1}) + V_{i+1} t_{i+1}] \quad \text{(A.10)}$$

$$\int_0^T V_u t^2 dt = \sum \frac{t_{i+1} - t_i}{2(3)} [V_i t_i^2 + 4V_{i+\frac{1}{2}} (\frac{t_i + t_{i+1}}{2})^2$$

$$+ V_{i+1} t_{i+1}^2]$$

$$= \sum \frac{t_{i+1} - t_i}{6} [V_i t_i^2 + V_{i+\frac{1}{2}} (t_i + t_{i+1})^2 + V_{i+1} t_{i+1}^2] \quad \text{(A.11)}$$

$$\int_0^T V_u t^3 dt = \sum \frac{t_{i+1} - t_i}{2(3)} [V_i t_i^3 + V_{i+\frac{1}{2}} (\frac{t_i + t_{i+1}}{2})^3$$

$$+ V_{i+1} t_{i+1}^3]$$

$$= \sum \frac{t_{i+1} - t_i}{6} [V_i t_i^3 + \frac{1}{2} V_i + \frac{1}{2} (t_i + t_{i+1})^3 + V_{i+1} t_{i+1}^3] \quad \text{(A.12)}$$

Equations (A.10) through (A.12) are evaluated in Boyce's Program.

1.4 Straight Line Correction - Trifunac's Program (Cal. Tech.)

To an acceleration term we apply a correction of the form

$$A_c = A_u - C_0 - C_1 t_i$$

where A_c is the corrected acceleration and A_u the uncorrected acceleration. We now wish to minimize $\int_0^T A_c{}^2 dt$ where T is the record length; thus:

$$\int_0^T A_c \frac{\delta A_c}{\delta C_0}\,dt = 0 \qquad \text{and} \qquad \int_0^T A_c \frac{\delta A_c}{\delta C_1}\,dt = 0$$

These two conditions give:

$$C_0 T + C_1 \frac{T^2}{2} = +\int A_u dt = A_1$$

$$C_0 \frac{T^2}{2} + C_1 \frac{T^3}{3} = +\int A_u t dt = A_2$$

We solve for the constants C_0 and C_1 by Cramer's Rule. The denominator becomes:

$$\begin{vmatrix} T & \frac{T^2}{2} \\ \frac{T^2}{2} & \frac{T^3}{3} \end{vmatrix} = \frac{T^4}{3} - \frac{T^4}{4} = \frac{T^4}{12}$$

The numerator becomes for C_0

$$\begin{vmatrix} A_1 & \frac{T^2}{2} \\ A_2 & \frac{T^3}{3} \end{vmatrix} = A_1 \frac{T^3}{3} - A_2 \frac{T^2}{2}$$

The numerator becomes for C_1

$$\begin{vmatrix} T & A_1 \\ \frac{T^2}{2} & A_2 \end{vmatrix} = A_2 T - A_1 \frac{T^2}{2}$$

Therefore:

$$C_0 = \frac{A_1 \frac{T^3}{3} - A_2 \frac{T^2}{2}}{\frac{T^4}{12}} = \frac{\frac{4}{3} A_1 T^3 - 2A_2 T^2}{\frac{T^4}{3}} \qquad (A.13)$$

$$C_1 = \frac{A_2 T - A_1 \frac{T^2}{2}}{\frac{T^4}{12}} = \frac{4A_2 T - 2A_1 T^2}{\frac{T^4}{3}} \qquad (A.14)$$

A_1 is the area under the acceleration-time curve (Figure A-2), or the velocity curve.

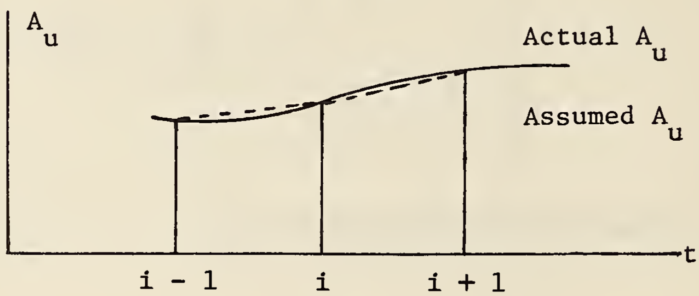


FIGURE A-2 ACCELERATION-TIME CURVE FOR NUMERICAL INTEGRATION

Suppose that the acceleration varies linearly between time stations. The acceleration between time t_i and t_{i+1} would then be approximated by:

$$A_u = A_{u(i)} + \frac{A_{u(i+1)} - A_{u(i)}}{t_{i+1} - t_i} (t - t_i) \qquad (A.15)$$

The velocity at any time within the same time interval may be obtained by:

$$V_u = V_{u(i)} + \int_{t_i}^{t} A_u dt$$

or

$$V_u = V_{u(i)} + A_{u(i)} (t - t_i) + \frac{A_{u(i+1)} - A_{i(i)}}{2(t_{i+1} - t_i)} (t - t_i)^2$$

which at station i + 1 becomes:

$$V_{u(i+1)} = V_{u(i)} + \frac{(t_{i+1} - t_i)}{2} (A_{u(i+1)} + A_{u(i)}) \qquad (A.16)$$

See line *LES 031* Appendix D

The displacement at t + 1 is given by:

$$X_{i+1} = X_i + \int_{t_i}^{t_{i+1}} V_u dt$$

$$= X_i + V_{u(i)}(t_{i+1}-t_i) + \frac{(t_{i+1}-t_i)^2}{6} (2A_{u(i)} + A_{u(i+1)}) \qquad (A.17)$$

See line *LES 030* Appendix D

A_2 may be evaluated within the same time interval as:

$$A_{2(i+1)} = A_{2(i)} + \int_{t_i}^{t} A_u \, t \, dt$$

$$= A_{2(i)} + \frac{A_{u(i)}}{2} (t_{i+1}{}^2 - t_i{}^2)$$

$$+ \frac{A_{u(i+1)} - A_{u(i)}}{3(t_{i+1} - t_i)} (t_{i+1}{}^3 - t_i{}^3)$$

$$\frac{-A_{u(i+1)} + A_{u(i)}}{2(t_{i+1} - t_i)} t_i (t_{i+1}{}^2 - t_i{}^2) \qquad (A.18)$$

However the Cal. Tech. program uses a simpler formula, which may be obtained by dropping the straight line approximation to the variation of acceleration and integrating $\int A_u(t) t \, dt$ over the entire length of the record by parts as an exact integral.

Let $u = t$, $du = dt$, $dv = A_u(t) dt$ and $\int dv = v = \int A_u(t) dt = V(t)$

Then using the method of parts:

$$\int_o^t A_u(t) t dt = t \times V(t) \Big]_o^T - \int_o^T V(t) dt$$

$$= T \times V(T) - 0 - \text{Final Displacement}$$

$$= \text{Final velocity} \times \text{Record Length (Time)} - \text{Final Displacement} \qquad (A.19)$$

See line *LES 43* Appendix D

Numerical calculations using equations (A.18) or (A.19) give the same results to three decimal places.

APPENDIX B

Filter Mathematics

Figure B-1 shows a desired response chart in a frequency domain from $-\omega_T$ to $+\omega_T$.

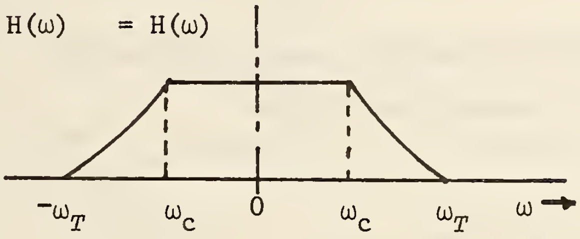


FIGURE B-1 RESPONSE CHART

It is desired to maintain a 1:1 ratio of input data to output data in the frequency range of $-\omega_c$ to $+\omega_c$. Let us therefore define a response function between $-\infty$ and $+\infty$ to be as follows:

$$H(\omega) = \begin{cases} 0 & ; \quad |\omega| > \omega_T \\ 1 & ; \quad |\omega| \leq \omega_c \\ \left(\frac{1}{\omega_T - \omega_c}\right)^P (\omega_T + \omega)^P & ; \quad -\omega_T \leq \omega < -\omega_c \\ \left(\frac{1}{\omega_T - \omega_c}\right)^P (\omega_T - \omega)^P & ; \quad \omega_c < \omega \leq \omega_T \end{cases} \tag{B.1}$$

where ω_c is the cut-off frequency.

The value of P determines the shape of the drop-off portions of the curve. If P = 1 the shape is a straight line, which the Cal. Tech. program assumes. For mathematics sake the folding frequency is taken as zero -- frequencies below zero are fictitious. The weight function h(t) associated with $H(\omega)$ is given by:

$$h(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{i\omega t}\, H(\omega)\, d\omega \qquad \text{(B.2)}$$

Noting that the integral can be broken into five distinct parts with appropriate limits,

$$h(t) = \frac{1}{2\pi}\int_{-\infty}^{-\omega_T} e^{i\omega t}\,(0)\,d\omega$$

$$+ \frac{1}{2\pi}\int_{-\omega_T}^{-\omega_c} e^{i\omega t}\left(\frac{1}{\omega_T-\omega_c}\right)^P (\omega_T+\omega)^P d\omega$$

$$+ \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} e^{i\omega t}\,(1)\,d\omega$$

$$+ \frac{1}{2\pi}\int_{\omega_c}^{\omega_T} e^{i\omega t}\left(\frac{1}{\omega_T-\omega_c}\right)^P (\omega_T-\omega)^P d\omega$$

$$+ \frac{1}{2\pi}\int_{\omega_T}^{\infty} e^{i\omega t}\,(0)\,d\omega \qquad \text{(B.3)}$$

the first and last term have zero factors, thus leaving only the second, third, and forth term to be integrated by parts.

Noting that $e^{i\omega t} = \cos \omega t + i \sin \omega t$ and letting P = 1, we evaluate the following equation term by term:

$$h(t) = \frac{1}{2\pi}\int_{-\omega_T}^{-\omega_c} e^{i\omega t}\left(\frac{\omega+\omega_T}{\omega_T-\omega_c}\right) d\omega + \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} e^{i\omega t} d\omega$$

$$+ \frac{1}{2\pi}\int_{\omega_c}^{\omega_T} e^{i\omega t}\left(\frac{\omega_T-\omega}{\omega_T-\omega_c}\right) d\omega \qquad (B.4)$$

Now:

$$\int_{-\omega_T}^{-\omega_c} e^{i\omega t}(\omega+\omega_T)\, d\omega = \int_{-\omega_T}^{-\omega_c} (\cos \omega t + i\sin\omega t)(\omega+\omega_T) d\omega$$

$$= \int \omega\cos\omega t d\omega + \int \omega_T\cos\omega t d\omega + \int i\omega\sin\omega t d\omega$$

$$+ \int i\omega_T\sin\omega t d\omega$$

where, using the method of parts

$$\int \omega\cos\omega t d\omega = \left[\frac{\omega\sin\omega t}{t}\right] - \frac{1}{t}\int \sin\omega t d\omega$$

$$= \frac{\omega_c\sin\omega_c t}{t} - \frac{\omega_T\sin\omega_T t}{t} + \frac{\cos\omega_c t}{t^2}$$

$$- \frac{\cos\omega_T t}{t^2} \qquad (B.5)$$

$$\int i\omega\sin\omega t d\omega = \left[-\frac{i}{t}\,\omega\cos\omega t\right] + \frac{i}{t}\int \cos\omega t d\omega$$

$$= \frac{i}{t}\,\omega_c\cos\omega_c t - \frac{i}{t}\omega_T\cos\omega\ t - \frac{i}{t^2}\sin\omega_c t$$

$$+ \frac{i}{t^2}\sin\omega_T t \qquad (B.6)$$

and

$$\int \omega_T \cos\omega t d\omega = \omega_T \sin\frac{\omega t}{t} = -\frac{\omega_T \sin\omega_c t}{t} + \frac{\omega_T \sin\omega_T t}{t} \qquad (B.7)$$

$$\int i\omega_T \sin\omega t d\omega = - i\omega \frac{\cos\omega t}{t} = -\frac{i\omega_T \cos\omega_c t}{t}$$

$$+ \frac{i\omega_T \cos\omega_T t}{t} \qquad (B.8)$$

Also:

$$\frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} e^{i\omega t} d\omega = \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} (\cos\omega t + i\sin\omega t)\, d\omega$$

$$= [\frac{1}{2\pi} \frac{\sin\omega t}{t}] - [\frac{i\cos}{2\pi} \frac{\omega t}{t}]$$

Where:

$$[\frac{i\cos\omega t}{2\pi t}]_{-\omega_c}^{\omega_c} = 0 \text{ and } [\frac{\sin\omega t}{2\pi t}]_{-\omega_c}^{\omega_c} = \frac{\sin\omega_c t}{\pi t} \qquad (B.9)$$

Finally:

$$\int_{\omega_c}^{\omega_T} e^{i\omega t} (\omega - \omega) d\omega = \int_{\omega_c}^{\omega_T} (\cos\omega t + i\sin\omega t)(\omega_T - \omega) d\omega$$

$$= \int \omega_T \cos\omega t d\omega = \int \omega\cos\omega t d\omega + \int i\omega_T \sin\omega t d\omega$$

$$- \int i\omega\sin\omega t d\omega$$

where, using the method of parts:

$$\int \omega\cos\omega t d\omega = - \frac{\omega\sin\omega t}{t} + \frac{1}{t} \int \sin\omega t d\omega$$

$$= -\frac{\omega_T \sin\omega_T t}{t} + \frac{\omega_c \sin\omega_c t}{t} - \frac{\cos\omega_T t}{t^2}$$

$$+ \frac{\cos\omega_c t}{t^2} \tag{B.10}$$

$$-\int i\omega \sin\omega t d\omega = \frac{i}{t}\,\omega\cos\omega t - \frac{i}{t}\int \cos\omega t d\omega$$

$$= \frac{i}{t}\,\omega_T \cos\omega\ t - \frac{i}{t}\ \omega_c \cos\omega_c t - \frac{i}{t^2}\sin\omega_T t$$

$$+ \frac{i}{t^2}\sin\omega_c t \tag{B.11}$$

and

$$\int \omega_T \cos\omega t d\omega = \omega_T \frac{\sin\omega t}{t} = \frac{\omega_T \sin\omega\ t}{t} - \frac{\omega_T \sin\omega_c t}{t} \tag{B.12}$$

$$\int i\omega_T \sin\omega t d\omega = -\ i\omega_T \cos\ \frac{\omega t}{t} = -\ \frac{i\omega_T \cos\omega_T t}{t}$$

$$+ \frac{i\omega_T \cos\omega_c t}{t} \tag{B.13}$$

Equation (B.6) cancels Equation (B.11)

Equation (B.8) cancels Equation (B.13)

Substitution of equations (B.5), (B.7), (B.9), (B.10), and (B.12) into equation (B.5) gives:

$$h(t) = \frac{1}{2\pi(\omega_T-\omega_c)}\ [2\cos\frac{\omega_c t}{t^2} - 2\cos\frac{\omega_T t}{t^2}$$

$$- \frac{2\sin\omega_c t}{t}(\omega_T-\omega_c)] + \frac{\sin\omega_c t}{\pi t}$$

$$= \frac{\cos\omega_c t - \cos\omega_T t}{\pi(\omega_T-\omega_c)t^2} \qquad \text{(B.14)}$$

The filter weights h(t) must be evaluated for equally spaced times. For use in the computer, we normalize the roll-off and cut-off frequencies (ω_T and ω_c) by denoting the variables $\lambda = \omega/\omega_s$, $\omega_c = 2\pi f_c$, $\omega_T = 2\pi f_t$, $\lambda_c = \omega_c/\omega_s$, and $\lambda_r = (\omega_T-\omega_c)/\omega_s$; where ω_s is the effective sampling angular frequency.

(note: $\Delta t = \frac{1}{f_s} = \frac{2\pi}{\omega_s} \qquad \omega_s = \frac{2\pi}{\Delta t}$

Assuming h(t) real, $H(\omega)$ turns out to be an even function; thus h_n = h-n and only n + 1 weights need be calculated. The following digitized non-dimensional formula results:

$$h_n = \frac{\cos 2\pi n\lambda_c - \cos 2\pi n\lambda_t}{2\lambda_r(\pi n)^2} \quad 1 \qquad n = 0, \pm 1, \pm 2, \ldots N \quad \lambda_t = \lambda_c + \lambda_r \qquad \text{(B-15)}$$

The formula is evaluated by the Ormsby subroutine. A special form of this equation needs to be evaluated for n = 0, when the formula becomes indeterminate using De l' Hospital's rule.

$$h_n = \frac{\cos(2\pi n\lambda_c) - \cos(2\pi n\lambda_t)}{2\lambda_r(\pi n)^2}$$

Let

$$h_n = \frac{g(n)}{f(n)}$$

$$\frac{\delta(g(n))}{\delta(f(n))} = - \frac{2\pi\lambda_c \sin(2\pi n\lambda_c) + 2\pi\lambda_t \sin(2\pi n\lambda_t)}{4\lambda_r \pi^2 n}$$

For n=0, $h_n = \infty$

Therefore use De l' Hospital's Rule again.

$$\frac{\delta^2(g(n))}{\delta^2(f(n))} = - \frac{4\pi^2\lambda_c^2 \cos(2\pi n\lambda_c) + 4\pi^2\lambda_t^2 \cos(2\pi n\lambda_t)}{4\lambda_r \pi^2}$$

$$= \frac{\lambda_t^2 \cos(2\pi n\lambda_t) - \lambda_c^2 \cos(2\pi n\lambda_c)}{\lambda_r}$$

For n=0, $\dfrac{\lambda_t^2 - \lambda_c^2}{\lambda_r} = \dfrac{(\lambda_t - \lambda_c)(\lambda_t + \lambda_c)}{\lambda_r}$

But $\lambda_r = \dfrac{\omega_T - \omega_c}{\omega_s} = \dfrac{\left(\dfrac{\omega_T - \omega_c}{\omega_s}\right)(\lambda_t + \lambda_c)}{\left(\dfrac{\omega_T - \omega_c}{\omega_s}\right)} = \lambda_t + \lambda_c$

The quantity $\lambda_r = (\omega_T - \omega_c)/\omega_s$ which specifies the sharpness of roll-off after λ_c together with the number of weights N

measures the resultant accuracy of $H(\lambda)$ with reduced accuracy for lower λ_r and/or lower N.

To provide a means for quickly determining the minimum N to choose for a decimal accuracy and sharpness of roll-off, a series of response runs were made by Ormsby (   ) covering a range of λ_c from 0 to 0.4, λ_r from 0.005 to 0.1 and N from 10 to 100. Figure B-2 shows the filter frequency response from his publication, for which he derived the following formula:

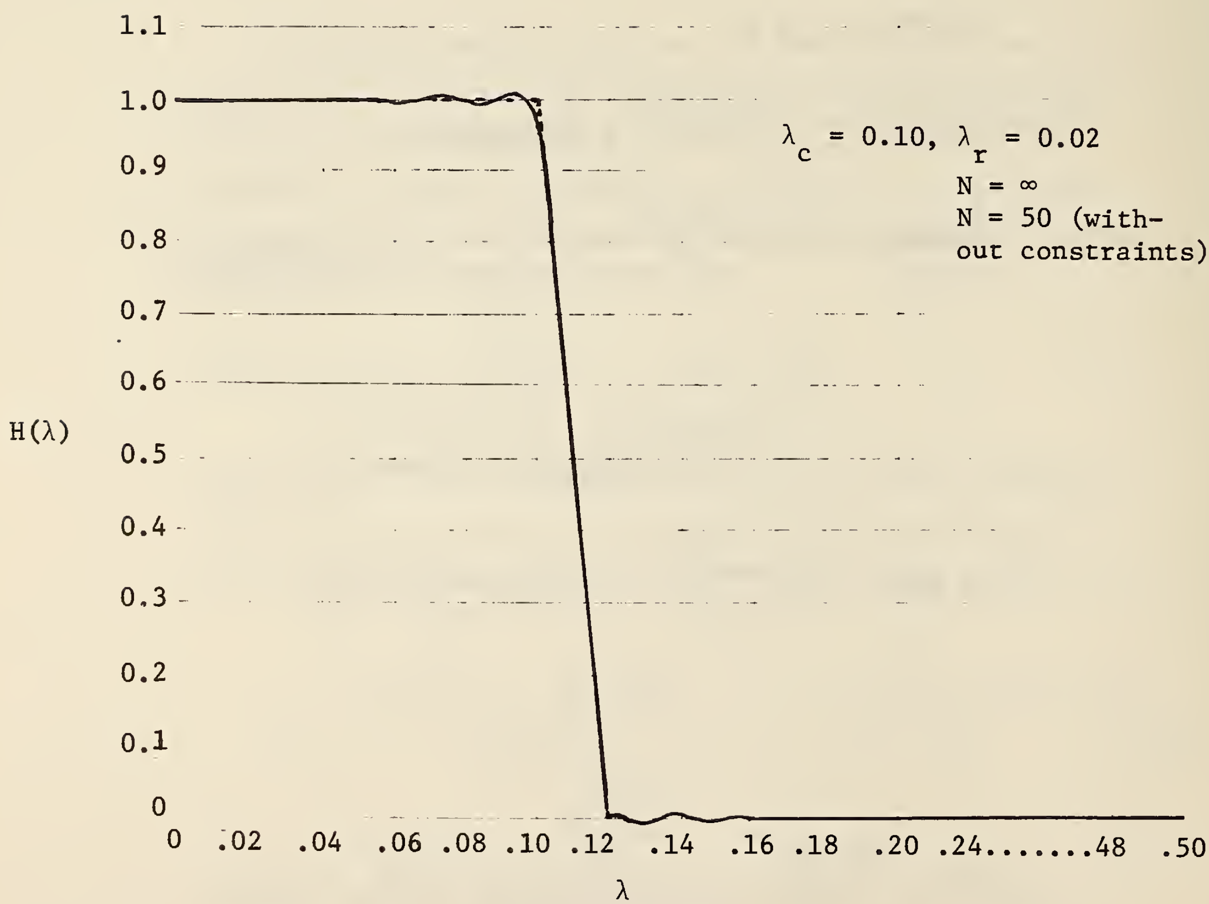


FIGURE B-2 FILTER FREQUENCY RESPONSE

$$\lambda_r N = \frac{0.012}{\varepsilon}$$

where ε is the tolerable error. Trifunac uses $\varepsilon = 0.012$, so that in effect

$$N = \frac{1}{\lambda_r}$$

in equation (B-15).

APPENDIX C

USER'S INSTRUCTIONS - PROGRAM DIASMA

1.1 PHASE1

The function of the PHASE1 subroutine is essentially that of taking raw digitized accelerogram data and converting it to data that can be used for subsequent integration and filtering. All the coordinate data (input and output) appears as two numbers which represent the time and acceleration values of an earthquake record. These digitized coordinates can be input via a 7 or 9-track magnetic tape or IBM computer cards. However, the title card and main control cards must be in punched card form. The output from PHASE1 is given as either a coordinate plot, a listing, or a punched deck of computer cards representing the desired output data.

On the output data deck, coordinate points are punched on 20 columns, with a maximum of four points per card. Each 20 columns are punched in the FORMAT 2F10.3, where the first 10 columns represent the time coordinate, and the second 10 columns represent the acceleration coordinate.

To utilize subroutine PHASE1, the following cards are required:

First Card:

Reduction factors for Calcomp Plotter.
FORMAT (3F10.3) XRED, YRED, DACCEL
This information is for limiting the final dimensions of the plot to any convenient size. For example, if XRED = 0.5 the plot is reduced to 1/2 normal size. If XRED = 2, the plot is doubled its normal size for the Time coordinate. YRED is the proportional size

change for the acceleration, velocity, or displacement coordinate. See page C-6 for DACCEL.

Second Card:

Determines the use of the program for digitization or integration.

FORMAT (2I5,4F10.6) NPHASE, NCORR, A, B, C, D

Column 5

NPHASE	= 1	Use will be for digitization.
	= 2	Use will be for integration.

Column 10

NCORR	= 0	Baseline corrected data will be prepared from corrected raw data.
	= 1	Raw data will be plotted to scale of original accelerogram.

Columns 11-20

A	=	Time scaling factor for PLOTTR subroutine.

Columns 21-30

B	=	Initial raw Time data to scale of A.

Columns 31-40

C	=	Acceleration scaling factor for PLOTTR subroutine.

Columns 41-50

D	=	Initial raw acceleration data to scale of C.

Third Card:

Title of accelerogram data

FORMAT (8A10)

Up to eighty characters can be used. Usually this card gives the location of the accelerograph and the component being digitized.

Fourth Card:

Main Control Card

FORMAT (3I5,4F10.0) NPLOT, NPUNCH, INPUTP, SIZE, XLEN, SCALET, SCALEA

Column 5

NPLOT	= 0	No plot is output.
	= 1	Input digitized data will be plotted.
	= 2	Baseline corrected and scaled data will be plotted.

(Note: Any number other than 1 or 2 will be interpreted as a zero, i.e., no plot.)

Column 10

NPUNCH	= 0	No cards will be punched.
	= 1	Input digitized data will be punched on cards.
	= 2	Baseline corrected and scaled data will be punched on cards.

(Note: Any number other than 1 or 2 will be interpreted as a zero, i.e., no punch.)

Column 15

INPUTP	= 1	Digitized coordinate values representing time and acceleration will be input via punched cards.
	= 2	Digitized coordinate values will be input via magnetic tape.

(Note: Any number other than 2 will be interpreted as a 1, i.e., computer expects input data to appear on punched cards.)

Columns 16-25

SIZE	=	Vertical distance (in inches) from the highest peak to the lowest valley of accelerogram trace.

Columns 26-35

XLEN = Horizontal distance (in inches) from the first time coordinate to the final time coordinate of accelerogram trace. XLEN is limited to positive values and the physical length of the plotting paper.

(Note: No value need be intered for SIZE and XLEN if NPLOT is specified as zero.)

Columns 36-45

SCALET = Time scaling factor used to convert digitized "counts" of time to any desired output units (e.g., seconds.)

Columns 46-55

SCALEA = Acceleration scaling factor used to convert digitized "counts" of acceleration to any desired output units (e.g., cm/sec^2 or units of gravity.)

(Note: No value need be inserted for SCALET or SCALEA if NPUNCH or NPLOT is any number other than 2. Also if INPUTP = 2, all subsequent data for this component must be on magnetic tape.)

Accelerogram Data Cards

The third card begins the digitized data deck (if INPUTP is specified as any number other than 2). The third and subsequent cards contain the digitized time and acceleration data.

FORMAT 4(2F10.0) TIME(1), ACCEL(1), . . .

Columns 1-10

TIME(1) = Digitized time coordinate of point 1.

Columns 11-20

ACCEL(1) = Digitized acceleration coordinate of point 1.

Columns 21-30

TIME(2) = Digitized time coordinate of point 2.

Columns 31-40

ACCEL(2) = Digitized acceleration coordinate of point 2.

And so forth . . .

Time and acceleration coordinate values are input in this manner; i.e., each coordinate point uses 20 columns to represent its location on the accelerogram. (Note: Four points i.e., pairs of coordinates must be digitized on each card; otherwise, blank spaces will be interpreted as zeros. The last data card may have fewer than four points.)

EOF Card:

An end-of-file card is placed after the last card containing accelerogram trace coordinates.

Fixed Trace Data Cards:

Fixed trace digitized data (if any) follows the EOF card.

FORMAT 4(2F10.0) X(I), Y(I),. . .

Digitized fixed trace data appears in the same as Accelerogram Data cards; i.e., four pairs of coordinate values per card representing the "X" and "Y" values of the fixed trace.

EOF Card:

An end-of-file card is placed after the last card containing digitized fixed trace coordinates.

(Note: The second EOF card is required even though there may be no fixed trace data cards.)

1.2 PHASE2

Subroutine PHASE2 does the job of processing earthquake data so that it may be acceptable for use in determining the design spectra for the earthquake. The input data appears in the same format as that of the PHASE1 program. That is, each digitized point is represented by a pair of numbers, the time and acceleration coordinates. The input data may be unscaled with no baseline correction or fixed trace correction. These operations can be performed in PHASE2. However, the subroutine has no provision for reading data from a magnetic tape. All input data must be in the form of punched cards.

The output from PHASE2 is in the form of a listing of the final filtered and integrated acceleration, velocity, and displacement, all as a function of time. The user may also elect to output a plotted graph of the final above information along with a graph of the major filter weights. No other form of output is provided.

To utilize the PHASE2 subroutine the following input cards are required:

First Card:

Reduction factors for Calcomp Plotter.

FORMAT (3F10.3) XRED, YRED

The first two items are for limiting the final dimension of the plot to any convenient size. For example, if XRED = 0.5 the plot is reduced to 1/2 normal size. If XRED = 2, the plot is doubled its normal size for the Time coordinate. YRED is the proportional size change for the acceleration, velocity, or displacement coordinate. DACCEL is the only non-general term in the computer listing, and was used to represent the maximum lift height of the pendulum in the experiments

described in this report. It was used to convert the measured digitized data to a zero baseline at the bottom position of the pendulum.

Second Card:

Determines the use of the program for digitization or integration.

FORMAT (2I5,4F10.6) NPHASE, NCORR, A, B, C, D

Column 5

NPHASE	= 1	Use will be for digitization.
	= 2	Use will be for integration.

Column 10

NCORR	= 0	Baseline corrected data will be prepared from corrected raw data.
	= 1	Raw data will be plotted to scale of original accelerogram.

Columns 11-20

A	=	Time scaling factor for PLOTTR subroutine.

Columns 21-30

B	=	Initial raw Time data to scale of A.

Columns 31-40

C	=	Acceleration scaling factor for PLOTTR subroutine.

Columns 41-50

D	=	Initial raw acceleration data to scale of C.

Third Card:

Title of accelerogram data

FORMAT (8A10)

Up to eighty characters can be used. Usually this card gives the location of the accelerograph and the component being digitized.

Fourth Card:

Main Control Card.

FORMAT (4F10.0,I5) T, CD, SCALET, SCALEA, NPLOT, ISHORT, NEWWAY, N2WAY, N3WAY, N4WAY

Columns 1-10

T	=	Natural period of the accelerograph transducer given in seconds.

Columns 11-20

CD	=	Critical damping coefficient of the accelerograph transducer given in percent.

(Note: This information is usually available from the seismiological station where the accelerogram was recorded. If this information is unknown and/or these first 20 columns left blank, the program will not perform an instrument correction on the data.)

Columns 21-30

SCALET	=	Time scaling factor used to convert input time data to seconds.

Columns 31-40

SCALEA	=	Acceleration scaling factor used to convert input acceleration data to units of G.

(Note: Program PHASE2 requires that the data is scaled to seconds and cm/sec/sec. Scaling factors that yield other units of time and acceleration will yield erroneous output. If the data is input in units of seconds and gravity, then SCALET and SCALEA should be defined as 1.0.)

Column 45

NPLOT	= 0	No plot is output.
	= 1	Final acceleration will be plotted.

= 2 Final acceleration, velocity, and displacement will be plotted.

= 3 Final acceleration, velocity, displacement and the low-pass filter weights will be plotted.

= 4 Only the low-pass filter weights will be plotted.

(Note: Any number other than the above will be interpreted as a zero, i.e., no plot.)

<u>Column 50</u>

ISHORT = 0 Instrument correction is performed.

= 1 No instrument correction is performed.

<u>Column 55</u>

NEWWAY = 0 Cal. Tech. integration procedure is used.

= 1 Horizontal baseline correction replaces leastsquare correction procedure. Filtering procedure follows Cal. Tech. method.

<u>Column 60</u>

N2WAY = 0 Cal. Tech. integration procedure is used.

= 1 Horizontal baseline correction for acceleration and velocity. No filter procedure for velocity or displacement.

<u>Column 65</u>

N3WAY = 0 Cal. Tech. integration procedure used.

= 1 Horizontal baseline correction replaces leastsquare procedure for acceleration and velocity. No filter procedure on acceleration, velocity, or displacement. HORIZ subroutine used on acceleration when data has been decimated to 0.02 sec.

Column 70
N4WAY = 0 Cal. Tech. integration procedure used.

= 1 Horizontal correction procedure for acceleration only. Filtering on acceleration and velocity, but not on displacement. The velocity is least-squared. Essentially this is the same as NEWWAY except that the displacement filter is eliminated. HORIZ subroutine used on acceleration when data has been decimated to 0.02 sec.

= 5 The Ormsby filter ramp is changed to $f_c = .48$, $f_t = .50$ Hz.

= 7 The Ormsby filter ramp is changed to $f_c = .08$, $f_t = .10$ Hz.

= 8 The Ormsby filter ramp is changed to $f_c = .28$, $f_t = .30$ Hz.

= 9 The Ormsby filter ramp is changed to $f_c = 1.98$, $f_t = 2.00$

= 10 The Ormsby filter ramp is changed to $f_c = .98$, $f_t = 1.0$ Hz. If, in addition N2WAY = 8, then an input displacement record will be high-pass filtered at 1 Hz.

Column 75
N5WAY = 1 HORIZ replaces the leastsquare subroutine for acceleration and velocity. No filters are used. Essentially this is the same as N3WAY except that HORIZ operates on the acceleration data while it is still interpolated at 0.01 sec.

Column 80
N6WAY = 1 HORIZ replaces the leastsquare subroutine for acceleration and velocity.

Filters are used for acceleration, velocity, and displacement.

Figure C-1 summarizes the above options from the Cal. Tech program by listing the various steps in each option.

Accelerogram Data Cards:

The third card begins the input data deck of the time-acceleration coordinates.

FORMAT 4(2F10.0) TIME(I), ACCEL(I), . . .

Each coordinate appears on 20 columns of a punched card with the time coordinate in the first 10 columns and the acceleration coordinate in the second 10 columns. All cards (except the last card) must contain four pairs of coordinate data. The decimal point may appear anywhere in the 10 columns or if it is omitted, it will be located after the digit in the tenth column of the coordinate.

EOF Card:

An end-of-file card is placed after the last card containing accelerogram trace coordinates.

STATE-MENT NO.	STEPS	CAL. TECH	NEWWAY = 1	N2WAY = 1	N3WAY = 1	N4WAY = 1	N5WAY = 1	N6WAY = 1
	Read Time & Accel	x	x	x	x	x	x	x
	Scale Time & Accel	x	x	x	x	x	x	x
	EQLSPC at .01 sec.	x	x	x	x	x	x	x
	HORIZ - on Accel for zero vel.						x	x
	ISHORT Call Ormsby							
	Decimate Accel at .02 sec.	x	x	x	x	x	x	x
	Instrument Correction							
130	HORIZ - on Accel for zero vel.		x	x	x	x		
	LESTSQ - on Accel - Subtract B - Get Accel5 at .02 sec.	x						
	HOLWAY - Accel6 .2	x	x	x		x		x
	ORMSBY - Accel8 .2	x	x	x		x		x
	EQLSPC Interpolate .02	x	x	x		x		x
270	Accel8 - 5 = 9	x	x	x		x		x
	HORIZ - Integrate Accel Get Vel.			x	x		x	x
	Integrate Accel9 - Then Leastsq. Vel. - Deduct B from Accel. - Correct Vel.	x	x			x		
330	Dec. Vel. to Vel 1 at .2	x	x			x		x
	ORMSBY on Vel. at .2	x	x			x		x
	Interp. at .02 Vel.	x	x			x		x
390	Vel. = Vel. - Vel 1	x	x			x		x
	Integrate Vel. to get Disp at .02	x	x	x	x	x	x	x
	Dec. Disp at .2 - Displ	x	x					x
	ORMSBY - Displ at .2	x	x					x
	Disp = Disp - Displ	x	x					x
	Convert Accel to Accel/G	x	x	x	x	x	x	x
470	PLOT	x	x	x	x	x	x	x

FIGURE C-1. SUMMARY OF PROGRAM STEPS FOR DIFFERENT OPTIONS FOR PROGRAM DIASMA

APPENDIX D

DIASMA Program Listing

Appendix D contains a listing of the computer program used by the authors to process the data of this report. The listing is given for both the IBM 360/370 O.S. computer and the CDC 6400 computer; together with control statements for running the program on either computer.

In the plot routine for PHASE2 decimation problems caused overruns beyond the ends of some arrays; therefore, the problem was solved by eliminating a few end points in the plot.

```
C
C     *******************************************************************
C
      IMPLICIT REAL*4 (A-H,O-Z)
C     PROGRAM DAISMA PROCESSES DIGITIZED STRONG MOTION EARTHOUAKE
C     ACCELERATION RECOROS TO OBTAIN GROUND VELOCITY ANO DISPLACEMENT.
C     THIS IS OONE IN TWO PHASES. PHASE 1 ACCEPTS OIGITIZED DATA FROM A
C     MAGNETIC TAPE OR COMPUTER CAROS, CHECKS THAT THE PLOT OF THE DATA
C     AGREES WITH THE ORIGINAL RECORO, THE DATA INCREASES MONOTONICALLY
C     WITH TIME, AND PUNCHES A NEW SET OF BASELINE CORRECTED DATA
C     STARTING AT TIME ZERO. PHASE 2 USES THE OUTPUT OF PHASE 1 TO
C     CONVERT THE OATA INTO UNITS OF CM FOR TIME AND CM/SEC/SEC FOR
C     ACCELERATION. FINALLY PHASE 2 INTEGRATES, LEASTSQUARES, AND
C     FILTERS THE DATA TO PRODUCE A PLOT OF GROUND ACCELERATION,
C     VELOCITY, AND DISPLACEMENT.
C
C         IF NCORR = 0, PHASE1 SUBROUTINE AUTOMATICALLY CORRECTS THE
C                       DIGITIZED DATA FOR MONITONICALLY INCREASING
C                       TIME.
C                  = 1, THE DIGITIZED OATA WILL BE HANO CORRECTEO ON THE
C                       KEYPUNCH AFTER THE PLOT OF THE DATA IS OBSERVED.
C                       FOR THIS CASE A,B,C,D ARE SCALING FACTORS TO
C                       CONVERT THE DIGITIZED DATA SO THAT THE PLOT WILL
C                       BE TO THE IOENTICAL SCALE AS THE ORIGINAL
C                       RECORO, I.E. COUNTS PER INCH OF PLOT.
C
C          A = SCALING FACTOR TO CONVERT TIME DATA TO PLOT SCALE
C          B = VALUE OF SCALED TIME DATA FOR TIME EQUALS ZERO
C          C = SCALING FACTOR TO CONVERT ACCELERATION DATA TO PLOT SCALE
C          D = VALUE OF SCALED ACCELERATION DATA FOR TIME EQUALS ZERO
C
C     *******************************************************************
C
C
      COMMON ACCEL(5500),TIME(5500),TITLE(20),VEL(3000),DISP(3000),LAST
      COMMON/PLTBUF/IBUF(1000)
      COMMON /ORMS/ WTS(275),NUMBER
      OIMENSION ACCEL1(5500),ACCEL2(5500),ACCEL3(5500),ACCEL4(5500),
     1 ACCEL6(5500),ACCEL7(5500),ACCEL8(5500),ACCEL9(5500)
      DIMENSION ATIME(5500),BTIME(5500),CTIME(5500),VEL1(300),
     1 DISP1(300),ATEMP(5500),ACCEL5(3000),TINST(3000)
      EQUIVALENCE (ACCEL,ACCEL1,ACCEL2,ACCEL3,ACCEL4,ACCEL6,ACCEL7,
     1 ACCEL8,ACCEL9)
      EQUIVALENCE (ACCEL5,VEL) , (TIME,ATIME,BTIME,CTIME,ATEMP)
      EQUIVALENCE (DISP1,VEL1), (LAST,NO1), (DISP,TINST)
      EQUIVALENCE (X,VEL) , (Y,DISP)
C
      REAO (1,1030) XREO,YRED,OACCEL
      WRITE (3,1040) XREO,YRED,DACCEL
C
   10 REAO (1,1000) NPHASE,NCORR,A,B,C,O
      WRITE(3,1010) NPHASE,NCORR,A,B,C,D
C
      IF(NPHASE .EQ. 1) CALL PHASE1(NCORR,A,B,C,D,XREO,YRED)
      IF(NPHASE .EQ. 2) CALL PHASE2(XREO,YRED,OACCEL)
C
 1000 FORMAT (2I5,4F10.6)
 1010 FORMAT (///
     1  17H NPHASE NUMBER = ,I5//
     1  17H          NCORR = ,I5//
     1  17H              A = ,F10.6//
     1  17H              B = ,F10.6//
     1  17H              C = ,F10.6//
     1  17H              D = ,F10.6//)
 1030 FORMAT (3F10.3)
 1040 FORMAT (1H1,10X,7HXREO = ,F10.3,5X,7HYREO = ,F10.3,9HDACCEL = ,
     .F10.3)
      ENO
```

```
      SUBROUTINE PHASE1(NCORR,A,B,C,D,XRED,YRED)                        PH1 001
      IMPLICIT REAL*4 (A-H,O-Z)
C                                                                       PH1 002
C       AN EOF CARD OR MARK (FOR MAG TAPE) SIGNALS END OF ACCELOGRAM OR PH1 003
C       FIXED TRACE (F. T.) DATA.  IF THERE IS NO F. T. DATA, PLACE     PH1 004
C       TWO (2) EOF CARD$ (MARKS) AT END OF ACCELOGRAM DATA.            PH1 005
C       IF THREE (3) EOF'S ARE ENCOUNTERED PER COMPONENT, PROGRAM STOPS: PH1 006
C                                                                       PH1 007
      COMMON ACCEL(5500),TIME(5500),TITLE(20),VEL(3000),DISP(3000),LAST
      COMMON/PLTBUF/IBUF(1000)
      DIMENSION X(1),Y(1)                                               PH1 009
      EQUIVALENCE (X,VEL),(Y,DISP)                                      PH1 010
C
C
C                                                                       PH1 013
   10 READ (1,2050,END=20) TITLE
      GO TO 30
   20 WRITE (3,2030)                                                    PH1 016
      WRITE (3,4000)
      STOP                                                              PH1 018
   30 WRITE (3,2060) TITLE
C                                                                       PH1 020
C                                                                       PH1 021
      READ (1,1050) NPLOT,NPUNCH,INPUTP,SIZE,XLEN,SCALET,SCALEA         PH1 022
      WRITE (3,3020)                                                    PH1 023
      WRITE (3,3010) NPLOT,NPUNCH,INPUTP,SIZE,XLEN,SCALET,SCALEA        PH1 024
C                                                                       PH1 025
C       NPLOT * = 0, NO PLOT IS DESIRED                                 PH1 026
C               = 1, DIGITIZED DATA WILL BE PLOTTED                     PH1 027
C               = 2, BASELINE CORRECTED DATA WILL BE PLOTTED            PH1 028
C                                                                       PH1 029
C       NPUNCH * = 0, NO CARDS WILL BE PUNCHED                          PH1 030
C                = 1, DIGITIZED DATA WILL BE PUNCHED ON CARDS           PH1 031
C                = 2, BASELINE CORRECTED DATA WILL BE PUNCHED ON CARDS  PH1 032
C                                                                       PH1 033
C       INPUTP *  = 1, DATA READ FROM PUNCHED CARDS                     PH1 034
C                 = 2, DATA READ FROM MAG TAPE                          PH1 035
C                                                                       PH1 036
C       SIZE = VERTICAL DISTANCE (IN INCHES) FROM HIGHEST PEAK TO       PH1 037
C       LOWEST VALLEY OF RECORD FROM WHICH RAW DATA IS DIGITIZED.       PH1 038
C       SIZE MUST NOT EXCEED TEN INCHES.                                PH1 039
C                                                                       PH1 040
C       XLEN = LENGTH OF TIME AXIS (IN INCHES) FROM TIME=0 TO MAX TIME  PH1 041
C              IN RECORD                                                PH1 042
C       XLEN SHOULD BE EQUAL TO ACTUAL RECORD LENGTH                    PH1 043
C       FROM WHICH DATA IS DIGITIZED                                    PH1 044
C                                                                       PH1 045
C                                                                       PH1 046
      IF (INPUTP .EQ. 1) WRITE (3,1010)                                 PH1 047
      IF (INPUTP .EQ. 2) WRITE (3,1020)                                 PH1 048
      IF (INPUTP .NE. 2) INPUTP = 1                                     PH1 049
      IF (NPLOT .EQ. 0) GO TO 40                                        PH1 050
      IF (NPLOT .EQ. 1) GO TO 40                                        PH1 051
C                                                                       PH1 052
      WRITE (3,1030) SIZE,XLEN                                          PH1 053
C                                                                       PH1 054
   40 IF (NPUNCH .EQ. 1) WRITE (3,1070)                                 PH1 055
      IF (NPUNCH .EQ. 2) WRITE (3,1060)                                 PH1 056
      IF (NPUNCH .EQ. 1 .OR. NPUNCH .EQ. 2) PUNCH 2050, TITLE
C                                                                       PH1 058
C     READ IN TIME-ACCEL DATA FROM DIGITIZED RECORDS                    PH1 059
C                                                                       PH1 060
      NT = -1                                                           PH1 061
      IF (INPUTP .EQ. 1) GO TO 85                                       PH1 062
C                                                                       PH1 063
C       MAG TAPE READING ROUTINE                                        PH1 064
C                                                                       PH1 065
      CALL TRK010                                                       PH1 066
```

```
C
      REWIND 7
      NT=0
   50 NT = NT+ 1
      IF (NT .LT. 2998) GO TO 70
   60 WRITE (3,2080) TIME(2998)
      GO TO 380
   70 READ (7,2040,END=80) TIME(NT), ACCEL(NT)
      GO TO 50
   80 LAST = NT
      GO TO 120
C
C       PUNCHED CARD READING ROUTINE
C
   85 NT=NT+1
      IF (NT .LT. 749) GO TO 100
   90 LAST = 4*NT
      WRITE (3,2070) TIME(LAST)
      STOP 1
  100 READ (1,1000,END=110) (TIME(NT*4+I),ACCEL(NT*4+I),I=1,4)
      GO TO 85
  110 LAST = 4*NT
  120 CONTINUE
      IF (ACCEL(LAST) .EQ. 0.0) LAST = LAST-1
      IF (ACCEL(LAST) .EQ. 0.0) LAST = LAST-1
      IF (ACCEL(LAST) .EQ. 0.0) LAST = LAST-1
      IF (NPUNCH .EQ. 1) PUNCH 1040, (TIME(I), ACCEL(I), I=1,LAST)
      WRITE (3,1090) LAST
      WRITE (3,1040) (TIME(I), ACCEL(I), I=1,LAST)
C
      IF (NCORR .EQ. 1) GO TO 150
C
C       CHECK DATA FOR CONTINUOUSLY INCREASING TIMES
C
      DO 140 M=2,LAST
      IF (TIME(M) -TIME(M-1)) 130,140,140
  130 TIME(M)=TIME(M-1)
  140 CONTINUE
C
C       READ IN FIXED TRACE DATA (IF ANY)
C
  150 NX = -1
      IF (INPUTP .EQ. 1) GO TO 180
      IF (NT .GE. 2998) GO TO 280
C
      CALL TRK010
C
      REWIND 7
C
      NX=0
  160 NX=NX+1
      READ (7,2040) X(NX),Y(NX)
      IF (EOF(7)) 170,160
  170 NFXTRC=NX
      GO TO 200
  180 NX=NX+1
      READ (1,1000,END=190) (X(NX*4+I), Y(NX*4+I), I=1,4)
      GO TO 180
  190 NFXTRC=4*NX
  200 CONTINUE
      IF (NFXTRC .LE. 1) GO TO 280
      IF (Y(NFXTRC) .EQ. 0.0) NFXTRC = NFXTRC - 1
      IF (Y(NFXTRC) .EQ. 0.0) NFXTRC = NFXTRC - 1
      IF (Y(NFXTRC) .EQ. 0.0) NFXTRC = NFXTRC - 1
      IF (NPUNCH .EQ. 1) PUNCH 1040, (X(I),Y(I),I=1,NFXTRC)
      WRITE (3,2000) NFXTRC
      WRITE (3,1040) (X(I), Y(I), I=1,NFXTRC)
C
```

```
C
      IF (NPLOT .EQ. 1) GO TO 380
C
C       SMOOTH FIXED TRACE DATA
C
      NFTM1 = NFXTRC-1
      DO 210 I=2,NFTM1
  210 Y(I) = Y(I-1)/4.0 + Y(I)/2.0 + Y(I+1)/4.0
C
C       DEDUCT FIX TRACE FROM ACCELEROGRAM
C
      IF (X(NFXTRC) .LT. TIME(LAST)) X(NFXTRC) = TIME(LAST)
      DO 220 J=1,LAST
      IF (TIME(J) .GT. X(1)) GO TO 230
  220 ACCEL(J)=ACCEL(J)-Y(1)
  230 JJ=J
      DO 270 I=1,NFXTRC
      DO 260 J=JJ,LAST
      IF (TIME(J) .GT. X(I+1)) GO TO 270
      IF (X(I+1)-X(I)) 250,250,240
  240 ACCEL(J)=ACCEL(J)-Y(I)-(TIME(J)-X(I))*(Y(I+1)-Y(I))/(X(I+1)-X(I))
      GO TO 260
  250 ACCEL(J) = ACCEL(J)-Y(I)
  260 CONTINUE
  270 JJ=J
  280 CONTINUE
      WRITE (3,2090)
      WRITE (3,1040) (TIME(I), ACCEL(I), I=1,LAST)
      IF (NCORR .EQ. 1) GO TO 380
      IF (NPUNCH .EQ. 1) GO TO 360
C
C       ADJUST DATA FOR ZERO BASELINE AND BEGINNING TIME = 0.0
C
      IF (SCALET .EQ. 0.0) SCALET = 1.0
      IF (SCALEA .EQ. 0.0) SCALEA = 1.0
      DO 290 I=2,LAST
  290 TIME(I)=(TIME(I)-TIME(1))*SCALET
      TIME(1)=0.0
      AREA = 0.0
      DO 300 I=2,LAST
      DT = TIME(I)-TIME(I-1)
  300 AREA = AREA+(ACCEL(I)+ACCEL(I-1))*DT/2.0
      ADJUST = AREA/TIME(LAST)
      WRITE (3,3000) ADJUST
      DO 310 I=1,LAST
  310 ACCEL(I) =(ACCEL(I) - ADJUST)*SCALEA
C
      IF (NFXTRC .GT. 1) GO TO 340
C
      AREA = 0.0
      DISP(1) = 0.0
      TLAST = TIME(LAST)
      DO 320 I=2,LAST
      DT = TIME(I)-TIME(I-1)
      DISP(I)=DISP(I-1)+AREA*DT+DT*DT/6.*(2.*ACCEL(I-1)+ACCEL(I))
  320 AREA=AREA+DT/2.*(ACCEL(I)+ACCEL(I-1))
      A=6./TLAST*DISP(LAST)/TLAST-2./TLAST*AREA
      B=6./TLAST*AREA/TLAST-2./TLAST*6./TLAST*DISP(LAST)/TLAST
      DO 330 I=1,LAST
  330 ACCEL(I)=ACCEL(I)-A-B*TIME(I)
  340 WRITE (3,1080)
      WRITE (3,1040) (TIME(I), ACCEL(I), I=1,LAST)
  350 IF (NPUNCH .NE. 2) GO TO 360
C
C       PUNCH BASELINE CORRECTED DATA
C
      WRITE (3,1060)
      PUNCH 1040, (TIME(I), ACCEL(I), I=1,LAST)
      WRITE (3,2010)
```

```
      WRITE (3,1040) (TIME(I), ACCEL(I), I=1,LAST)
  360 CONTINUE
      IF (NPLOT .EQ. 0) GO TO 450
      WRITE (3,2020)
      WRITE (3,1040) (TIME(I), ACCEL(I), I=1,LAST)
      XMIN = TIME(1)
      TIME(LAST+1) = XMIN
      XMAX = TIME(LAST)
      SCALX =(XMAX-XMIN)/XLEN
      TIME(LAST+2) = SCALX
      YMIN = ACCEL(1)
      YMAX = ACCEL(1)
      DO 370 I=2,LAST
      IF (ACCEL(I) .GT. YMAX) YMAX = ACCEL(I)
      IF (ACCEL(I) .LT. YMIN) YMIN = ACCEL(I)
  370 CONTINUE
      ACCEL(LAST+1) = YMIN
      ACCEL(LAST+2) = (YMAX-YMIN)/SIZE
      SCALY = (YMAX-YMIN)/SIZE
      WDY = SIZE+0.5
      IF (WDY .GT. 10.0) WDY = 10.0
C
C       PLOT TIME - ACCELERATION DATA
C
  380 CALL PLOTS(IBUF,1000,5)
      IF (XRED .EQ. 1.0 .AND. YRED .EQ. 1.0) GO TO 390
      CALL SETFACT (XRED,YRED)
  390 IF (NCORR .EQ. 1) GO TO 420
      CALL PLOT (0.,-11.,-3)
      CALL PLOT (0.,0.5,-3)
      WRITE(3,9998)
 9998 FORMAT(1H ,10X,"BEFORE CALL AXIS")
      CALL AXIS(0.0,0.0,"REL. TIME IN COUNTS",-19,XLEN,0.,XMIN,SCALX)
      CALL AXIS(0.0,0.0,"REL. ACCEL. IN COUNTS",21,SIZE,90.,YMIN,SCALY)
      WRITE (3,9999)
 9999 FORMAT(1H ,10X,"AFTER CALL AXIS")
      CALL SYMBOL (0.5,WDY,0.25,TITLE,0.,80)
      IF (LAST .LE. 200) GO TO 410
  400 J=0
      ISYM=0
      GO TO 430
  410 J=1
      ISYM=4
      IF(NPLOT.EQ.2) GO TO 430
      X(NFXTRC+1)=TIME(LAST+1)
      X(NFXTRC+2)=TIME(LAST+2)
      Y(NFXTRC+1)=ACCEL(LAST+1)
      Y(NFXTRC+2)=ACCEL(LAST+2)
      GO TO 430
C
  420 CALL PLOT (0.,-12.,-3)
      CALL PLOT (0.,5.5,-3)
      CALL SYMBOL (0.,0.,1.,3,0.,-1)
      CALL PLOTTR (TIME,ACCEL,LAST,A,B,C,D)
      CALL PLOTTR (X,Y,NFXTRC,A,B,C,D)
      CALL SYMBOL (0.5,6.0,0.25,TITLE,0.,80)
      GO TO 440
  430 CALL LINE (TIME,ACCEL,LAST,1,J,ISYM)
      IF(NPLOT.EQ.2) GO TO 440
      CALL LINE (X,Y,NFXTRC,1,J,ISYM)
  440 CALL PLOT(20.0,0.0,450)
C
  450 CONTINUE
C
C
 1000 FORMAT (4(2F10.0))
 1010 FORMAT (/,10X,48HTIME - ACCELERATION DATA READ FROM PUNCHED CARDS
     1)
 1020 FORMAT (/,10X,43HTIME - ACCELERATION DATA READ FPOM MAG TAPE/)
```

```
1030 FORMAT (/,10X,41HBASELINE CORRECTED DATA WILL BE PLOTTED -,F5.1,
    116H INCHES HIGH AND,F5.1,12H INCHES LONG)
1040 FORMAT (4(2F10.3))
1050 FORMAT ( 3I5,4F10.0)
1060 FORMAT (10X,48HBASELINE CORRECTED DATA WILL BE PUNCHED ON CARDS/)
1070 FORMAT (10X,58HDIGITIZED DATA WILL BE PUNCHED ON CARDS EXACTLY AS
    1FOLLOWS/)
1080 FORMAT (1X,/,10X,31HBASELINE CORRECTED DATA FOLLOWS/)
1090 FORMAT (10X,21HNO OF DATA POINTS ARE,I5,/,10X,38HDATA AS IT APPEAR
    1S FROM INPUT FOLLOWS //)
2000 FORMAT (/10X,28HNO OF FIXED TRACE POINTS ARE,I5,34H AND APPEARS F
    1OM INPUT AS FOLLOWS//)
2010 FORMAT (1X,/,10X,20HPUNCHED DATA FOLLOWS//)
2020 FORMAT (1X,//,10X,20HPLOTTED DATA FOLLOWS//)
2030 FORMAT (2X,//,10X,32HNO MORE DATA, PROGRAM TERMINATES)
2040 FORMAT (4X,2(F6.0))
2050 FORMAT (20A4)
2060 FORMAT (1H,20A4,//)
2070 FORMAT (1X,/,10X,58HINPUT DATA EXCEEDED ARRAY SIZE - REMOVE DATA
    1EYOND TIME =,F10.3,18H AND RERUN PROGRAM//)
2080 FORMAT (1X,/,10X,57HINPUT DATA EXCEEDED ARRAY SIZE - DATA TRUNCAT
    1D AT TIME =,F10.3,/,10X,42HFIXED TRACE DATA -IF ANY- WILL NOT BE
    2SED//)
2090 FORMAT (1X,/,10X,45HSUBTRACTED FIXED TRACE DATA FROM ACCELEROGRAM/
    .)
3000 FORMAT (1X,/,9HADJUST = ,F10.3,/)
3010 FORMAT (10X,I1,4X,I1,5X,I1,5X,4F10.3,/)
3020 FORMAT (1X,/,10X,5HNPLOT,6HNPUNCH,6HINPUTP,6X,4HSIZE,6X,4HXLEN,4X,
    .6HSCALET,4X,6HSCALEA)
4000 FORMAT (10X,31HNORMAL TERMINATION, END OF DATA)
     RETURN
     END


      SUBROUTINE TRK010
C
C***** THIS SUBROUTINE READS DATA FROM AN ELECTRAK DATA TAPE AND OUTPUTS
C       ON TAPE IN DATA WHICH IS IN A FORM ACCESSIBLE BY PHASE1 PROGRAM.
C      THIS PROGRAM IS A PATTERN MATCH PROCEDURE, PROBABLY
C       BETTER SUITED TO WRITING IN SNOBOL4.  THE TARGET PATTERN IS
C      ZZZZZZZNNNNSNNNNNSNNNNNCCCCCCCCCCCCCCCCCCCZZZZ...
C
C      WHERE  Z IS ANY CHARACTER
C             N IS ANY DECIMAL DIGIT
C             S IS A SIGN (+ OR -)
C             C IS A SET OF CHARACTERS ASSOCIATED WITH THE NUMBER
C      THE GOAL IS TO OUTPUT RECORDS OF THE ENTIRE 36 CHARACTER
C      RECORD AND THE COMMENTS BETWEEN THIS AND THE PRECEEDING
C      PATTERN.  THESE RECORDS ARE OUTPUT ON TAPE7.
C
      IMPLICIT INTEGER (A-Z)
      LOGICAL*1 JBUF(6000),JPLUS(4),NPLUS,JMINUS(4),MINUS,CHR(4)
      LOGICAL*1 LPLUS(4),LMINUS(4),CHR2(4)
      DIMENSION IBUF(1500)
      EQUIVALENCE (IPLUS,JPLUS),(IMINUS,JMINUS),(INT,CHR)
      EQUIVALENCE (KPLUS,LPLUS),(KMINUS,LMINUS),(INT2,CHR2)
      DATA IPLUS/1H+/,IMINUS/1H-/,LBUF/1260/
      DATA CHRWRD/4/,END/11/,NEXT/6/,BEGIN/4/
      DATA IN/6/,OUT/7/,PRT/3/
C
C     SET UP POINTERS AND STUFF
C

      LAST=0
      REWIND OUT.
```

```
      MINUS=JMINUS(1)
      NPLUS=JPLUS(1)
      KPLUS=0
      KMINUS=0
      LPLUS(4)=JPLUS(1)
      LMINUS(4)=JMINUS(1)
      NPRU=0
      WRITE (PRT,1)
    1 FORMAT (1H1)
C
C     READ A DATA BLOCK AND MOVE TO CHARACTER ARRAY
C
   10 READ (IN,11,END=998,ERR=996) (IBUF(I),I=1,LBUF)
   11 FORMAT (10(126A4))
      NPRU=NPRU+1
   20 CONTINUE
      LENGTH=LBUF
      DO 30 I=1,LENGTH
      INT=IBUF(I)
      DO 40 J=1,CHRWRD
      LAST=LAST+1
      JBUF(LAST)=CHR(J)
   40 CONTINUE
   30 CONTINUE
      POS=1
      INT=0
      INT2=0
C
C     SEARCH THROUGH BLOCK FOR TARGET PATTERN
C
   50 CONTINUE
      IF (POS+END .GT. LAST) GO TO 100
      CHR2(4)=JBUF(POS+NEXT)
      CHR(4)=JBUF(POS)
      IF(((INT.EQ.KPLUS).OR.(INT.EQ.KMINUS))
     .  .AND. ((INT2.EQ.KPLUS) .OR. (INT2.EQ.KMINUS)))
     .    GO TO 60
      POS=POS+1
      GO TO 50
C
C     PATTERN HAS BEEN FOUND
C
   60 CONTINUE
      J=POS-BEGIN
      K=POS+END
      WRITE (OUT,70) (JBUF(I),I=J,K)
   70 FORMAT (16A1)
      WRITE (PRT,80) (JBUF(I),I=J,K)
   80 FORMAT (1X,16A1)
      POS=K+1
      GO TO 50
C
C     NOT ENOUGH DATA FOR THE NEXT PATTERN MATCH, REPACK DATA
C
  100 CONTINUE
      J=0
      DO 110 I=POS,LAST
      J=J+1
      JBUF(J)=JBUF(I)
  110 CONTINUE
      LAST=J
      GO TO 10
C
C     PARITY ERROR HAS BEEN DETECTED
C
```

```
  996 CONTINUE
      NPRU=NPRU+1
      WRITE (PRT,997) NPRU
  997 FORMAT (40H *** PARITY ERROR DETECTED IN PRU NUMBER,I5,4H ***)
      GO TO 20
C
C     END OF FILE OR ERROR HAS BEEN DETECTED
C
  998 CONTINUE
      WRITE (PRT,999)
  999 FORMAT (29H *** END OF FILE DETECTED ***)
      REWIND OUT
      RETURN
      END
C
C
C
C


      SUBROUTINE PLOTTR (X,Y,LAST,A,B,C,D)
      IMPLICIT REAL*4 (A-H,O-Z)
C
      DIMENSION X(3000),Y(3000)
C
      X(1)=A*X(1)-B
      Y(1)=C*Y(1)-D
      CALL PLOT (X(1),Y(1),3)
      DO 1 I=2,LAST
      X(I)=A*X(I)-B
      Y(I)=C*Y(I)-D
    1 CALL SYMBOL (X(I),Y(I),0.1,74,0.0,-1)
      RETURN
      END


      SUBROUTINE PHASE2(XRED,YRED,DACCEL)
      IMPLICIT REAL*4 (A-H,O-Z)
C
C     *****************************************************************
C
C      SUBROUTINE PHASE2 IS A COMPUTER PROGRAM WHICH READS DIGITIZED
C      ACCELEROGRAM DATA AND PROCESSES THIS DATA IN ORDER TO ELIMINATE
C      ERRORS IN RECORDING AND DIGITIZING THE ACCELEROGRAM.
C      THE CORRECTED ACCELOGRAM IS NUMERICALLY INTEGRATED TO OBTAIN THE
C      CORRESPONDING VELOCITY AND DISPLACEMENT.
C      THESE QUANTITIES ARE NOW ACCEPTABLE FOR USE IN DETERMINING
C      THE DESIGN SPECTRA FOR THE EARTHQUAKE.
C      FINALLY, A PLOT OF THESE QUANTITIES WILL BE PLOTTED IF DESIRED.
C
C     *****************************************************************
C
      COMMON ACCEL(5500),TIME(5500),TITLE(20),VEL(3000),DISP(3000),LAST
      COMMON/PLTBUF/IBUF(1000)
      COMMON /ORMS/ WTS(275),NUMBER
      DIMENSION ACCEL1(5500),ACCEL2(5500),ACCEL3(5500),ACCEL4(5500),
     1 ACCEL6(5500),ACCEL7(5500),ACCEL8(5500),ACCEL9(5500)
      DIMENSION ATIME(5500),BTIME(5500),CTIME(5500),VEL1(300),
     1 DISP1(300),ATEMP(5500),ACCEL5(3000),TINST(3000)
      EQUIVALENCE (ACCEL,ACCEL1,ACCEL2,ACCEL3,ACCEL4,ACCEL6,ACCEL7,ACCEL
     18,ACCEL9)
      EQUIVALENCE (ACCEL5,VEL), (TIME,ATIME,BTIME,CTIME,ATEMP)
      EQUIVALENCE (DISP1,VEL1), (LAST,NO1), (DISP,TINST)
```

```
C
C
C
      READ (1,1090) TITLE
      WRITE (3,2000) TITLE
C
C       - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
C
C       T = NATURAL PERIOD OF ACCELEROMETER IN SEC.
C       CD = DAMPING COEFFICIENT OF ACCELEROMETER IN PER CENT OF CRITICAL
C       SCALET = TIME SCALING FACTOR TO CONVERT TIME DATA TO SECONDS
C       SCALEA = ACCELERATION SCALING FACTOR TO CONVERT ACCELERATION
C                DATA TO UNITS OF GRAVITY
C       NPLOT * = 0, NO PLOT IS DESIRED
C               = 1, PLOT ACCELERATION ONLY
C               = 2, PLOT ACCEL,VELOCITY, AND DISPLACEMENT
C               = 3, PLOT ALL THE ABOVE AND THE ORMSBY FILTER CORRECTION
C               = 4, PLOT ORMSBY FILTER ONLY
C
C       - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
C
      READ (1,1030) T,CD,SCALET,SCALEA,NPLOT,ISHORT,NEWWAY,N2WAY,N3WAY,
     .N4WAY,N5WAY,N6WAY
      WRITE (3,4000)
      WRITE (3,1060) T,CD,SCALET,SCALEA,NPLOT,ISHORT,NEWWAY,N2WAY,N3WAY,
     .N4WAY,N5WAY,N6WAY
C
C       READ IN TIME-ACCEL DATA FROM DIGITIZED RECORDS
C
      CALL REDATA (SCALET)
C
C       CONVERT DATA TO SEC AND CM/SEC/SEC (ACCEL1)
C
      CALL DATALT (SCALET,SCALEA)
C
      FRQMIN = 0.07
      FRQMAX = 25.0
C
C       NO1 = MAXIMUM NUMBER OF 0.01 SECOND INTERVALS + 1
C
      N1 = LAST
      NO1 = (TIME(LAST)+0.009)*100.0
      LAST = NO1
      NO2 = (LAST+1)/2
      N = NO2+9
      N2 = N/10
      N2P1 = N2+1
      N2M1 = N2 - 1
      NO2 = 10 * (N2-1) + 1
      NNO2 = NO2-8
      WRITE (3,2019)
      WRITE (3,2020) NO1,NO2,N1,N2
C
C
C       STORE UNEQUALLY SPACED TIME DATA TEMPORARILY IN DISP ARRAY.
C
      DO 10 I=1,N1
   10 DISP(I)=TIME(I)
C
C       OBTAIN INTERPOLATED VALUES (ACCEL2) AT 0.01 SEC TIME INCREMENTS
C
      DELT=0.01
C
      CALL EQLSPC (DELT,N1)
C
      IF (N5WAY .EQ. 1) GO TO 18
      IF(N5WAY .EQ. 2) GO TO 40
      IF (N6WAY .EQ. 1) GO TO 18
      GO TO 20
```

```
   18 NPASS = 5
      CALL HORIZ(NPASS)
   20 CONTINUE
C
C       IF ACCELEROGRAM WAS NOT RECORDED BY AN ACCELEROGRAPH, DO NOT
C       LOW-PASS FILTER
      IF (CD .LE. 0.0 .OR. T .LE. 0.0) GO TO 40
C
C       APPLY ORMSBY LOW-PASS FILTER (ACCEL3)
C
      FSUBC = FRQMAX
      FSUBT = FSUBC+2.0
      DELT=0.01
      ISYM = 1
C
      NSHORT = 1
      CALL ORMSBY (ISYM,FSUBC,FSUBT,DELT,ACCEL3,NSHORT)
      NSHORT = 0
C
C       DISCARD EVERY OTHER POINT OF SMOOTHED CURVE
C
      L=1
      DO 30 I=1,LAST,2
      ACCEL4(L) = ATEMP(I)
      BTIME(L)=(L-1)*0.02
   30 L=L+1
      BTIME(L)=(L-1)*0.02
C
C       EQUALLY SPACED POINTS ARE NOW 0.02 SEC APART (ACCEL4)
C
      WRITE (3,4010)
      WRITE (3,2080)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=1,8)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=NNO2,NO2)
C
      GO TO 70
   40 L = 1
      DO 50 I=1,LAST,2
      ACCEL4(L)= ACCEL2(I)
      BTIME(L)=(L-1)*0.02
   50 L=L+1
      BTIME(L)=(L-1)*0.02
   60 CONTINUE
C
      IF(N5WAY .EQ. 2) N5WAY = 1
      WRITE (3,4010)
      WRITE (3,2080)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=1,8)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=NNO2,NO2)
   70 CONTINUE
      IF (N2WAY .EQ. 8) GO TO 72
      GO TO 78
   72 DO 73 I=1,NO2
   73 ACCEL(I) = ACCEL(I) - DACCEL
C
      DO 74 I=1,NO2
   74 DISP(I) = ACCEL(I)
      GO TO 405
   78 IF (T .LE. 0.0 .OR. CD .LE. 0.0) GO TO 110
C
C       CORRECT FOR INSTRUMENT RESPONSE TO OBTAIN ABSOLUTE GROUND ACCEL.
C
   90 DELT = 0.02
      WRITE (3,1040) T,CD
      WO = 6.28318531/T
      CD = CD*0.01
      NXM = NO2-1
      DO 100 I=2,NXM
```

```
      D1=(ACCEL4(I+1)-ACCEL4(I-1))/(2.0*DELT)
      D2=(ACCEL4(I-1)-2.0*ACCEL4(I)+ACCEL4(I+1))/DELT**2
  100 TINST(I)=(D2 + 2.0*WO*CD*D1 + ACCEL4(I)*WO**2)/WO**2
      TINST(1) = ACCEL4(1)
      TINST(NO2) = ACCEL4(NO2)
      DO 105 I=1,NO2
  105 ACCEL(I) = TINST(I)
      WRITE (3,1050)
      WRITE (3,2080)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=1,8)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=NNO2,NO2)
C
      GO TO 120
  110 WRITE (3,2010)
C
  120 CONTINUE
C
C       LEAST SQUARE SMOOTHED CURVE AND SAVE (ACCEL5)
C
      NPASS = 1
C
      IF (NEWWAY .EQ. 1) GO TO 130
      IF (N2WAY  .EQ. 1) GO TO 130
      IF (N3WAY  .EQ. 1) GO TO 130
      IF (N4WAY  .EQ. 1) GO TO 130
      IF (N4WAY .EQ. 5) GO TO 150
      IF (N5WAY  .EQ. 1) GO TO 270
      IF (N6WAY  .EQ. 1) GO TO 190
      GO TO 150
  130 NPASS = 6
      CALL HORIZ (NPASS)
      GO TO 170
  150 CALL LESTSQ (ACCEL5,NPASS)
      IF (N4WAY .EQ. 5) GO TO 278
  160 CONTINUE
C
      LO2 = NO2 + 1
  170 DO 180 I=1,LO2
  180 ACCEL(I) = VEL(I)
      IF (N3WAY .EQ. 1) GO TO 270
C
C
C       APPLY RUNNING MEAN FILTER (ACCEL5)
C
  190 CALL HOLWAY (NO2)
C
      WRITE (3,2070)
      WRITE (3,2080)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=1,8)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=NNO2,NO2)
C
C
C     DECIMATE ACCELERATION FOR LOW PASS FILTERING
      L=1
      DO 210 I = 1,NO2,10
      ACCEL7(L) = ACCEL6(I)
      CTIME(L)=(L-1)*0.2
  210 L=L+1
      CTIME(L)=(L-1)*0.2
C
C       EQUALLY SPACED POINTS ARE NOW 0.2 SEC APART (ACCEL7)
C
C
      WRITE (3,3090)
      WRITE (3,3000)
      WRITE (3,1010) (CTIME(I), ACCEL7(I), I=1,N2P1)
C
C       APPLY ORMSBY LOW-PASS FILTER ON DECIMATED DATA
C
```

```
      FSUBC = FROMIN-0.02
      FSUBT = FROMIN
      IF (NEWWAY .EQ. 4) FSUBT = 2.00
      IF (NEWWAY .EQ. 4) FSUBC = 1.95
      DELT=0.2
      ISYM = 1
C
      CALL ORMSBY (ISYM,FSUBC,FSUBT,DELT,ACCEL7,NSHORT)
C
C      FILTERED DATA IS PLACED IN ACCEL ARRAY
C
      DO 230 I=1,N2P1
      ACCEL8(I)=ATEMP(I)
  230 TINST(I)=(I-1)*0.2
      WRITE (3,3020)
      WRITE (3,3070)
      WRITE (3,1010) (TINST(I), ACCEL8(I), I=1,N2P1)
C
C     INTERPOLATE TO 0.02 SEC TIME INTERVALS
C
      DELT=0.02
C
      CALL EQLSPC (DELT,N2M1)
C
      WRITE (3,3080)
      WRITE (3,2080)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=1,8)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=NN02,N02)
C
C
C      SUBTRACT LOW-PASS FILTERED DATA (ACCEL8) FROM LEAST SQ DATA
C      (ACCEL5) AND RESTORE FILTERED ACCELERATION DATA IN ACCEL9 ARRAY.
C
      DO 260 I=1,N
  260 ACCEL9(I)=ACCEL5(I)-ACCEL8(I)
  270 IF (N2WAY .EQ. 1) WRITE (3,2030)
      IF (N2WAY .EQ. 1) GO TO 275
      IF (N3WAY .EQ. 1) WRITE (3,2030)
      IF (N3WAY .EQ. 1) GO TO 275
      IF (N5WAY .EQ. 1) WRITE (3,2030)
      IF (N5WAY .EQ. 1) GO TO 275
      WRITE (3,4060)
  275 WRITE (3,2080)
      WRITE (3,1010) (TIME(I),ACCEL9(I),I=1,8)
      WRITE (3,1010) (TIME(I),ACCEL9(I),I=NN02,N02)
C
C
C     CALL LESTSQ TO INTEGRATE NEW ACCEL9 DATA AND GET VELOCITY.
C      THEN LEAST SQ VEL AND DEDUCT CORRECTION TERM (B) FROM ACCEL9.
C
  278 NPASS = 2
      IF (N2WAY .EQ. 1) GO TO 280
      IF (N3WAY .EQ. 1) GO TO 280
      IF (N4WAY .EQ. 5) GO TO 290
      IF (N5WAY .EQ. 1) GO TO 280
      IF (N6WAY .EQ. 1) GO TO 280
      IF (N6WAY .EQ. 6) GO TO 280
      GO TO 290
  280 CALL HORIZ (NPASS)
      GO TO 300
  290 CALL LESTSQ (ACCEL9,NPASS)
      WRITE (3,4050)
  300 CONTINUE
      IF (N2WAY .EQ. 1) GO TO 390
      IF (N3WAY .EQ. 1) GO TO 390
      IF (N5WAY .EQ. 1) GO TO 390
      WRITE (3,2030)
C
```

```
      WRITE (3,2080)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=1,8)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=NNO2,NO2)
C
C
      WRITE (3,2050)
      WRITE (3,3030)
      WRITE (3,1010) (TIME(I),VEL(I), I=1,8)
      WRITE (3,1010) (TIME(I),VEL(I), I=NNO2,NO2)
C
  330 CONTINUE
C
C
C       DECIMATE VELOCITY FOR LOW-PASS FILTERING
C
      L=1
      DO 340 I=1,NO2,10
      VEL1(L)=VEL(I)
  340 L=L+1
      WRITE (3,4020)
      WRITE (3,3040)
      WRITE (3,1010) (TIME(I),VEL1(I), I=1,N2P1)
C
C       EQUALLY SPACED DATA ARE NOW 0.2 SEC APART
C
C       APPLY ORMSBY LOW PASS FILTER
C
      DELT=0.2
      IF (N4WAY .EQ. 5) FSUBT = 0.5
      IF (N4WAY .EQ. 5) FSUBC = 0.48
      IF (N4WAY .EQ. 7) FSUBT = 0.1
      IF (N4WAY .EQ. 7) FSUBC = 0.08
      IF (N4WAY .EQ. 8) FSUBT = 0.3
      IF (N4WAY .EQ. 8) FSUBC = 0.28
      IF (N4WAY .EQ. 9) FSUBT = 2.0
      IF (N4WAY .EQ. 9) FSUBC = 1.98
      IF (N4WAY .EQ. 10) FSUBT = 1.0
      IF (N4WAY .EQ. 10) FSUBC = 0.98
C
      CALL ORMSBY (ISYM,FSUBC,FSUBT,DELT,VEL1,NSHORT)
C
C
C       FILTERED DATA IS PLACED IN VEL1 ARRAY
C
      DO 360 I=1,N2
  360 VEL1(I)=ATEMP(I)
      WRITE (3,4030)
      WRITE (3,4080)
      WRITE (3,1000) (I,VEL1(I), I=1,N2P1)
C
C       INTERPOLATE BACK TO 0.02 SEC TIME INTERVALS
C
      DELT = 0.02
      NPASS = 1
C
      CALL INTERP (DELT,N2,NPASS,VEL1,ATEMP,NEWWAY)
C
C       EQUALLY SPACED POINTS ARE NOW 0.02 SEC APART
C
C       SUBTRACT FILTERED VELOCITY (ATEMP) FROM INITIAL VELOCITY (VEL).
C
      DO 380 I=1,NO2
      VEL(I)=VEL(I)-ATEMP(I)
  380 TIME(I)=(I-1)*0.02
      WRITE (3,4070)
      WRITE (3,2030)
      WRITE (3,2080)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=1,8)
      WRITE (3,1010) (TIME(I),ACCEL(I),I=NNO2,NO2)
```

```
C
  390 CONTINUE
      WRITE (3,1070)
      WRITE (3,3030)
      WRITE (3,1010) (TIME(I),VEL(I), I=1,8)
      WRITE (3,1010) (TIME(I),VEL(I), I=NNO2,NO2)
C
C
C       INTEGRATE NEW VELOCITY (VEL) TO OBTAIN NEW DISPLACEMENT (DISP).
      IF (N2WAY .EQ. 1) NO2 = NO2 + 1
C
      DT = 0.02
      DISP(1)=0.0
      NM1 = NO2-1
      DO 400 I=1,NM1
  400 DISP(I+1) = DISP(I) + (VEL(I) + VEL(I+1))*DT/2.0
      IF (N4WAY .EQ. 5) GO TO 405
      IF (N2WAY .EQ. 1) WRITE (3,1080)
      IF (N3WAY .EQ. 1) WRITE (3,1080)
      IF (N4WAY .EQ. 1) WRITE (3,1080)
      IF (N5WAY .EQ. 1) WRITE (3,1080)
      WRITE (3,4090)
      WRITE (3,4040)
      WRITE (3,1010) (TIME(I),DISP(I),I=1,8)
      WRITE (3,1010) (TIME(I),DISP(I),I=NNO2,NO2)
C
      IF (N2WAY .EQ. 1) GO TO 470
      IF (N3WAY .EQ. 1) GO TO 470
      IF (N4WAY .EQ. 1) GO TO 470
      IF (N5WAY .EQ. 1) GO TO 470
  405 CONTINUE
C
C       DECIMATE DISPLACEMENT FOR LOW PASS FILTERING.
C
      L=1
      DO 410 I=1,N,10
      DISP1(L)=DISP(I)
  410 L=L+1
C
C       EQUALLY SPACED DATA ARE NOW 0.2 SEC APART
C
      IF (N4WAY .EQ. 5) FSUBT = 0.5
      IF (N4WAY .EQ. 5) FSUBC = 0.48
      IF (N4WAY .EQ. 7) FSUBT = 0.1
      IF (N4WAY .EQ. 7) FSUBC = 0.08
      IF (N4WAY .EQ. 8) FSUBT = 0.3
      IF (N4WAY .EQ. 8) FSUBC = 0.28
      IF (N4WAY .EQ. 9) FSUBT = 2.0
      IF (N4WAY .EQ. 9) FSUBC = 1.98
      IF (N4WAY .EQ. 10) FSUBT = 1.0
      IF (N4WAY .EQ. 10) FSUBC = 0.98
C       APPLY ORMSBY LOW PASS FILTER
C
      ISYM = 0
      DELT=0.2
C
      CALL ORMSBY (ISYM,FSUBC,FSUBT,DELT,DISP1,NSHORT)
C
C       FILTERED DATA IS PLACED IN DISP1 ARRAY
C
      DO 420 I=1,N2P1
  420 DISP1(I)=ATEMP(I)
      WRITE (3,3010)
      WRITE (3,3060)
      WRITE (3,1000) (I,DISP1(I), I=1,N2P1)
C
C       INTERPOLATE BACK TO 0.02 SEC TIME INTERVALS
C
      DELT = 0.02
```

```
      NPASS = 0                                                         PH2 438
C                                                                       PH2 439
      CALL INTERP (DELT,N2,NPASS,DISP1,ATEMP,NEWWAY)                    PH2 440
C                                                                       PH2 441
C       EQUALLY SPACED POINTS ARE NOW 0.02 SEC APART                    PH2 442
C                                                                       PH2 443
C       SUBTRACT FILTERED DISPLACEMENT (ATEMP) FROM INITIAL DISPL. (DISP) PH2 444
C                                                                       PH2 445
      DO 440 I=1,NO2                                                    PH2 446
  440 DISP(I)=DISP(I)-ATEMP(I)                                          PH2 447
      WRITE (3,1080)                                                    PH2 448
C                                                                       PH2 449
      DO 450 I=1,NO2                                                    PH2 450
  450 TIME(I)=(I-1)*0.02                                                PH2 451
      WRITE (3,4040)                                                    PH2 452
      WRITE (3,1010) (TIME(I), DISP(I), I=1,NO2)                        PH2 453
  470 CONTINUE                                                          PH2 454
C                                                                       PH2 455
C
      IF (N2WAY .EQ. 8) GO TO 475                                       PH2 457
C
C
      IF (NPLOT .LE. 0 .OR. NPLOT .GE. 5) GO TO 500                     PH2 460
C                                                                       PH2 461
C       CONVERT ACCELERATION BACK TO UNITS OF GRAVITY                   PH2 462
C                                                                       PH2 463
  480 CONTINUE                                                          PH2 464
      DO 490 I=1,N                                                      PH2 465
  490 ACCEL9(I)=ACCEL9(I)/980.665                                       PH2 466
C                                                                       PH2 467
C                                                                       PH2 468
      WRITE (3,2060)                                                    PH2 469
      WRITE (3,2080)                                                    PH2 470
      WRITE (3,1010) (TIME(I),ACCEL9(I),I=1,8)                          PH2 471
      WRITE (3,1010) (TIME(I),ACCEL9(I),I=NNO2,NO2)                     PH2 472
C                                                                       PH2 473
      DELT = 0.02                                                       PH2 474
      CALL PLTDAT (NPLOT,DELT,XRED,YRED,NO2)                            PH2 475
C                                                                       PH2 476
  500 CONTINUE                                                          PH2 477
C                                                                       PH2 478
 1000 FORMAT((1X,6(I5,FI0.3)))                                          PH2 479
 1010 FORMAT((1X,8(F8.3,F9.3)))                                         PH2 480
 1020 FORMAT((1X,6(F10.3,E10.3)))                                       PH2 481
 1030 FORMAT (4F10.6,8I5)                                               PH2 482
 1040 FORMAT (1X,/,10X,31HNATURAL PERIOD OF INSTRUMENT IS,F10.5,/,10X,  PH2 483
     1 45HCRITICAL DAMPING COEFFICIENT OF INSTRUMENT IS,F10.5,8H PERCENT PH2 484
     2//)                                                               PH2 485
 1050 FORMAT (1X,/,IOX,36HAFTER INSTRUMENT RESPONSE CORRECTION//)       PH2 486
 1060 FORMAT (IOX,4F10.6,8I10/)                                         PH2 487
 1070 FORMAT (1X,/,10X,24HFINAL VELOCITY IN CM/SEC/)                    PH2 488
 1080 FORMAT (1X,/,10X,24HFINAL DISPLACEMENT IN CM/)                    PH2 489
 1090 FORMAT (20A4)
 2000 FORMAT (1H,20A4,//)
 2010 FORMAT (/,10X,41HNO INSTRUMENT RESPONSE CORRECTION IS MADE/)      PH2 492
 2019 FORMAT (1X,/,3X,3HNO1,2X,3HNO2,3X,2HN1,3X,2HN2)                   PH2 493
 2020 FORMAT (1X,4I5)                                                   PH2 494
 2030 FORMAT (1X,/,10X,32HFINAL ACCELERATION IN CM/SEC/SEC/)            PH2 495
 2040 FORMAT (1X,/,10X,26HACCELERATION IN CM/SEC/SEC/)                  PH2 496
 2050 FORMAT (1X,/,10X,18HVELOCITY IN CM/SEC/)                          PH2 497
 2060 FORMAT (IX,/,10X,23HFINAL ACCELERATION IN G/)                     PH2 498
 2070 FORMAT (1X,/,10X,30HHOLLOWAY FILTERED ACCELERATION/)              PH2 499
 2080 FORMAT (5X,4HTIME,4X,5HACCEL)                                     PH2 500
 3000 FORMAT (4X,5HCTIME,3X,6HACCEL7)                                   PH2 501
 3010 FORMAT (IX,/,10X,35HORMSBY FILTERED DISPLACEMENT, DISP1/)         PH2 502
 3020 FORMAT (1X,/,10X,45HFILTERED DATA PLACED IN ACCEL ARRAY AT .2 SEC/ PH2 503
     )                                                                  PH2 504
 3030 FORMAT (5X,4HTIME,6X,3HVEL)                                       PH2 505
```

```
3040 FORMAT (5X,4HTIME,5X,4HVEL1)                                       PH2 506
3060 FORMAT (5X,1HI,5X,5HDISP1)                                         PH2 507
3070 FORMAT (4X,5HTINST,3X,6HACCEL8)                                    PH2 508
3080 FORMAT (1X,/,10X,32HFILTERED ACCELERATION AT .02 SEC/)             PH2 509
3090 FORMAT (1X,/,10X,41HDECIMATED ACCELERATION PRIOR TO FILTERING/)    PH2 510
4000 FORMAT (16X,1HT,9X,2HCD,5X,6HSCALET,5X,6HSCALEA,9X,5HNPLOT,        PH2 511
   .        5X,6HISHORT,4X,6HNEWWAY,4X,5HN2WAY,5X,5HN3WAY,5X,5HN4WAY,   PH2 512
   .        5X,5HN5WAY,5X,5HN6WAY/)                                     PH2 513
4010 FORMAT (1X,/,10X,51HEQUALLY SPACED POINTS ARE NOW .02 SEC APART(AC PH2 514
   .CEL4)/)                                                             PH2 515
4020 FORMAT (1X,/,10X,36HDECIMATED VELOCITY, VEL1, AT .2 SEC./)         PH2 516
4030 FORMAT (1X,/,10X,41HORMSBY FILTERED VELOCITY, VEL1, AT .2 SEC/)    PH2 517
4040 FORMAT (5X,4HTIME,5X,4HDISP)                                       PH2 518
4050 FORMAT (1X,/,10X,50HACCEL9 WITH B TERM FROM LESTSQ VELOCITY SUBTRA PH2 519
   .CTED/)                                                              PH2 520
4060 FORMAT (1X,/,10X,24HACCEL9 = ACCEL5 - ACCEL8/)                     PH2 521
4070 FORMAT (1X,/,10X,57HACCEL9 CORRECTED FOR VELOCITY CHANGE DUE TO OR PH2 522
   .MSBY FILTER/)                                                       PH2 523
4080 FORMAT (5X,1HI,6X,4HVEL1)                                          PH2 524
4090 FORMAT (1X,/,10X,23HINTEGRATED DISPLACEMENT/)                      PH2 525
     RETURN                                                             PH2 526
     END                                                                PH2 527


     SUBROUTINE REDATA (A)                                              RED 001
     IMPLICIT REAL*4 (A-H,O-Z)
C                                                                       RED 002
C     THIS SUBROUTINE READS DIGITIZED DATA FROM PUNCHED CARDS AND       RED 003
C     CHECKS FOR INCREASING VALUES OF TIME                              RED 004
C                                                                       RED 005
     COMMON ACCEL(5500),TIME(5500),TITLE(20),VEL(3000),DISP(3000),LAST
C
C
C                                                                       RED 009
     NT=-1                                                              RED 010
  10 NT=NT+1                                                            RED 011
     READ (1,1000,END=20) (TIME(NT*4+I),ACCEL(NT*4+I), I=1,4)
     GO TO 10
  20 LAST = 4*NT                                                        RED 014
C                                                                       RED 015
     IF (ACCEL(LAST) .EQ. 0.0) LAST = LAST-1                            RED 016
     IF (ACCEL(LAST) .EQ. 0.0) LAST = LAST-1                            RED 017
     IF (ACCEL(LAST) .EQ. 0.0) LAST = LAST-1                            RED 018
     IF (LAST .LE. 2998) GO TO 40
C                                                                       RED 020
  30 WRITE (3,1040) TIME(2998)                                          RED 021
     STOP 777                                                           RED 022
  40 XN = TIME(LAST)*A                                                  RED 023
     IF (XN .LE. 54.95) GO TO 60
C                                                                       RED 025
  50 XNN = 54.95/A                                                      RED 026
     WRITE (3,1040) XNN                                                 RED 027
C                                                                       RED 028
     WRITE (3,2000)
C                                                                       RED 030
     STOP 77                                                            RED 031
  60 WRITE (3,1030) LAST                                                RED 032
C                                                                       RED 033
C     CHECK DATA FOR CONTINUOUSLY INCREASING TIMES                      RED 034
C                                                                       RED 035
     GAP = 1.0/A/4.0                                                    RED 036
C                                                                       RED 037
C     GAP = 0.25 SEC IN TERMS OF INPUT TIME VALUES                      RED 038
C     DATA POINTS DIGITIZED FURTHER APART THAN THIS VALUE ARE NOT       RED 039
C     CONTINUOUS AND THE REMAINING DATA IS TRUNCATED.                   RED 040
C     IF THE DIFFERENCE BETWEEN TWO ADJACENT VALUES OF TIME IS LESS     RED 041
```

```
C       THAN GAP, AND THE TIME VALUES ARE DECREASING, THE SMALLER VALUE   RED 042
C       IS EQUATED TO THE PREVIOUS LARGER VALUE.                           RED 043
C                                                                          RED 044
      LM1 = LAST-1                                                         RED 045
C                                                                          RED 046
      DO 80 M=1,LM1                                                        RED 047
      IF (ABS(TIME(M+1)-TIME(M)) .GT. GAP) GO TO 90                        RED 048
      IF(TIME(M+1)-TIME(M)) 70,80,80                                       RED 049
   70 TIME(M+1) = TIME(M)                                                  RED 050
   80 CONTINUE                                                             RED 051
C                                                                          RED 052
      GO TO 100                                                            RED 053
   90 IF (M .LT. LAST) WRITE (3,1020) GAP, M                               RED 054
      LAST = M                                                             RED 055
  100 CONTINUE                                                             RED 056
      LLAST = LAST - 8                                                     RED 057
      WRITE (3,1050)                                                       RED 058
      WRITE (3,1010) (TIME(I), ACCEL(I), I=1,8)                            RED 059
      WRITE (3,1010) (TIME(I), ACCEL(I), I=LLAST,LAST)                     RED 060
C
C                                                                          RED 062
 1000 FORMAT (4(2F10.0))                                                   RED 063
 1010 FORMAT((1X,8(F8.3,F9.3)))                                            RED 064
 1020 FORMAT (10X,68HTHE MAXIMUM DIFFERENCE ALLOWED BETWEEN SUCCESSIVE V  RED 065
     1ALUES OF TIME IS ,F10.3,//,10X,81HTHIS DIFFERENCE HAS BEEN EXCEEDE  RED 066
     2D, THUS THE TOTAL NUMBER OF POINTS ARE REDUCED TO,I5//)              RED 067
 1030 FORMAT (10X,27HNO OF INPUT DATA POINTS ARE,I5//)                     RED 068
 1040 FORMAT(//,10X,13HTOO MUCH DATA,//,10X,26HREMOVE DATA BEYOND TIME    RED 069
     2=,F10.3,18H AND RERUN PROGRAM//)                                     RED 070
 1050 FORMAT (5X,4HTIME,4X,5HACCEL)                                        RED 071
 2000 FORMAT (10X,28HINPUT DATA OVERLOADS PROGRAM)
      RETURN                                                               RED 072
      END                                                                  RED 073


      SUBROUTINE DATALT (A,B)                                              DAT 001
      IMPLICIT REAL*4 (A-H,O-Z)
C                                                                          DAT 002
C       THIS SUBROUTINE CONVERTS THE DATA TO USABLE QUANTITIES             DAT 003
C                                                                          DAT 004
      COMMON ACCEL(5500),TIME(5500),TITLE(20),VEL(3000),DISP(3000),LAST
C

C
C                                                                          DAT 008
C                                                                          DAT 009
      IF (A .EQ. 1.0 .AND. B .EQ. 1.0) GO TO 20                            DAT 010
      WRITE (3,1020) A,B                                                   DAT 011
C                                                                          DAT 012
      DO 10 I=1,LAST                                                       DAT 013
      TIME(I) = TIME(I)*A                                                  DAT 014
      ACCEL(I) = ACCEL(I)*B*980.665                                        DAT 015
   10 CONTINUE                                                             DAT 016
C                                                                          DAT 017
      GO TO 40                                                             DAT 018
   20 CONTINUE                                                             DAT 019
      DO 30 I=1,LAST                                                       DAT 020
   30 ACCEL(I)=ACCEL(I)*980.665                                            DAT 021
   40 CONTINUE                                                             DAT 022
   50 CONTINUE                                                             DAT 023
C                                                                          DAT 024
C       CORRECT TIME BASE TO BEGIN WITH INITIAL TIME = 0                   DAT 025
C                                                                          DAT 026
      IF(TIME(1) .EQ. 0.0) GO TO 70                                        DAT 027
      DO 60 I=2,LAST                                                       DAT 028
      TIME(I)=TIME(I)-TIME(1)                                              DAT 029
   60 CONTINUE                                                             DAT 030
      TIME(1)=0.0                                                          DAT 031
```

```
   70 CONTINUE
      LLAST = LAST - 8
      WRITE (3,1000)
      WRITE (3,1030)
      WRITE (3,1010) (TIME(I), ACCEL(I), I=1,8)
      WRITE (3,1010) (TIME(I), ACCEL(I), I=LLAST,LAST)
C
C
 1000 FORMAT (15X,79HREVISED DATA WITH TIME IN SEC - ACCELERATION IN CM/
     1SEC/SEC - BASELINE CORRECTED//)
 1010 FORMAT((1X,8(F8.3,F9.3)))
 1020 FORMAT (//,10X,27HTIME ARRAY IS MULTIPLIED BY,F10.6,18H TO OBTAIN
     1SECONDS,/,10X,35HACCELERATION ARRAY IS MULTIPLIED BY,F10.6,47H TO
     2OBTAIN ACCELERATION IN TERMS OF GRAVITY (G)//)
 1030 FORMAT (5X,4HTIME,4X,5HACCEL)
      RETURN
      END



      SUBROUTINE EQLSPC (DELT,N1)
      IMPLICIT REAL*4 (A-H,O-Z)
C
C      FROM THE ACCEL ARRAY IN COMMON, THE DATA IS INTERPOLATED TO
C      EQUALLY SPACED TIMES (DELT) AND REPLACES OLD UNEQUALLY SPACED
C      DATA IN ACCEL ARRAY
C      THE TEMP ARRAY TEMPORARILY STORES THE DATA DURING INTERPOLATION.
C
      COMMON ACCEL(5500),TEMP(5500),TITLE(20),VEL(3000),TIME(3000),LAST
C
C
C
C     N = NUMBER OF EQUALLY SPACED TIME INCREMENTS
      N = LAST
      IF (DELT .EQ. 0.02) N=N1*10
      M=N+1
      NN = N1 + 1
      WRITE (3,1020) N
C
      LL=1
      ATIME = 0.0
      DO 30 I=2,NN
      DO 20 L=LL,M
      IF (ATIME .GT. TIME(I)) GO TO 30
      IF (TIME(I) -TIME(I-1)) 30,30,10
   10 TEMP(L)=ACCEL(I-1)+(ACCEL(I)-ACCEL(I-1))*(ATIME-TIME(I-1))/(TIME(I
     1)-TIME(I-1))
      ATIME = L*DELT
   20 CONTINUE
   30 LL = L
      DO 40 I=2,M
      ACCEL(I)=TEMP(I)
   40 TEMP(I)=(I-1)*DELT
      TEMP(1)=0.0
C
      WRITE (3,1000)
      LM = M - 8
      WRITE (3,1010) (TEMP(I), ACCEL(I), I=1,8)
      WRITE (3,1010) (TEMP(I), ACCEL(I), I=LM,M)
C
C
 1000 FORMAT (7X,4HTIME,5X,5HACCEL)
 1010 FORMAT((1X,4(2F10.3)))
 1020 FORMAT (1X,/,10X,27HEQUAL SPACING ROUTINE - N =,I5//)
      RETURN
      END
```

```
      SUBROUTINE LESTSQ (RET,NPASS)                                     LES 001
      IMPLICIT REAL*4 (A-H,O-Z)
C                                                                       LES 002
C      TEMP ARRAY IS ALWAYS LEAST SQUARED.                              LES 003
C      RETN ARRAY RETURNS PROPER DATA  AND SHARES STORAGE WITH ACCEL5.  LES 004
C                                                                       LES 005
      COMMON ACCEL(5500),TIME(5500),TITLE(20),VEL(3000),TEMP(3000),LAST
      DIMENSION RET(3000)                                               LES 007
C
C
C                                                                       LES 010
      DT = 0.02                                                         LES 011
      TLAST = FLOAT(LAST)*0.01                                          LES 012
      IF (TLAST .EQ. 0.0) TLAST = FLOAT(LAST-1)*0.01                    LES 013
      N = LAST/2                                                        LES 014
      NP1 = N+1                                                         LES 015
      DO 10 I=1,NP1                                                     LES 016
      IF (TIME(I) .LT. 0.0) GO TO 20
   10 CONTINUE                                                          LES 018
C                                                                       LES 019
   20 NBEG = I                                                          LES 020
      WRITE (3,1080) NP1                                                LES 021
C                                                                       LES 022
      IF (NPASS .EQ. 2) GO TO 110                                       LES 023
   30 AREA = 0.0                                                        LES 024
      DIS = 0.0                                                         LES 025
      NM1 = N-1                                                         LES 026
C                                                                       LES 027
      DO 40 I=1,N                                                       LES 028
      IF (I .EQ. NBEG) AREA1 = AREA                                     LES 029
      IF (I .EQ. NBEG) DIS1 = DIS                                       LES 030
      DT = TIME(I+1)-TIME(I)                                            LES 031
      DIS=DIS+AREA*DT+DT*DT/6.0*(2.0*ACCEL(I)+ACCEL(I+1))               LES 032
   40 AREA=AREA+(ACCEL(I)+ACCEL(I+1))*DT/2.0                            LES 033
C                                                                       LES 034
      WRITE (3,2030)                                                    LES 035
      WRITE (3,1090) DIS,AREA,N,TLAST,TIME(N+1)                         LES 036
C                                                                       LES 037
      DENOM=(1./3.)*TLAST**4                                            LES 038
C                                                                       LES 039
C      CHECK FOR DIVISION BY ZERO                                       LES 040
C                                                                       LES 041
      IF (DENOM .NE. 0.0) GO TO 60
   50 WRITE (3,1060)                                                    LES 043
      GO TO 70                                                          LES 044
   60 A =(4./3.)*(AREA-AREA1)*TLAST**3-2.*(AREA*TLAST-DIS+DIS1)*TLAST**2 LES 045
C                                                                       LES 046
      A = A/DENOM                                                       LES 047
      B = 4.0*TLAST*(AREA*TLAST-DIS+DIS1)-2.0*(AREA-AREA1)*TLAST**2     LES 048
      B = B/DENOM                                                       LES 049
      WRITE (3,2040)                                                    LES 050
      WRITE (3,1070) NPASS,A,B                                          LES 051
C                                                                       LES 052
   70 IF (NPASS .GE. 2) GO TO 140                                       LES 053
C                                                                       LES 054
C     LEAST SQ FIT ACCEL(I)                                             LES 055
C                                                                       LES 056
      DO 80 I=1,NP1                                                     LES 057
      CORREC=A+B*TIME(I)                                                LES 058
   80 RET(I) = ACCEL(I) - CORREC                                        LES 059
      NNP1 = NP1 - 8                                                    LES 060
      WRITE (3,1000)                                                    LES 061
      WRITE (3,2020)                                                    LES 062
      WRITE (3,1010) (TIME(I), RET(I), I=1,8)                           LES 063
      WRITE (3,1010) (TIME(I), RET(I), NNP1,NP1)                        LES 064
C
```

```
C                                                                        LES 066
C      LEAST SQUARED VALUES NOW IN RET ARRAY.                            LES 067
C                                                                        LES 068
      NPASS = 3                                                          LES 069
C                                                                        LES 070
C      INTEGRATE TO GET VELOCITY.                                        LES 071
C                                                                        LES 072
  110 CONTINUE                                                           LES 073
      TEMP(1)=0.0                                                        LES 074
      DO 120 I=1,N                                                       LES 075
      DT = TIME(I+1)-TIME(I)                                             LES 076
  120 TEMP(I+1)=TEMP(I)+(RET(I)+RET(I+1))*DT*0.5                         LES 077
      LN = N - 8                                                         LES 078
      WRITE (3,1020)                                                     LES 079
      WRITE (3,2000)                                                     LES 080
      WRITE (3,1010) (TIME(I), TEMP(I), I=1,8)                           LES 081
      WRITE (3,1010) (TIME(I), TEMP(I), LN,N)                            LES 082
C
C                                                                        LES 084
C      AT THIS POINT, THE TEMP ARRAY CONTAINS VELOCITY.                  LES 085
C                                                                        LES 086
      GO TO 30                                                           LES 087
  140 DO 150 I=1,NP1                                                     LES 088
  150 RET(I)=RET(I)-B                                                    LES 089
      WRITE (3,1040)                                                     LES 090
      WRITE (3,2010)                                                     LES 091
      WRITE (3,1010) (TIME(I), RET(I), I=1,8)                            LES 092
      WRITE (3,1010) (TIME(I), RET(I), NNP1,NP1)                         LES 093
C
      IF (NPASS .NE. 2) RETURN                                           LES 095
      DO 170 I=1,NP1                                                     LES 096
      ACCEL(I) = RET(I)                                                  LES 097
      VEL(I)= TEMP(I)-A-B*TIME(I)                                        LES 098
  170 CONTINUE                                                           LES 099
C                                                                        LES 100
      WRITE (3,1050)                                                     LES 101
      WRITE (3,2000)                                                     LES 102
      WRITE (3,1010) (TIME(I), VEL(I), I=1,16)                           LES 103
 1000 FORMAT (1X,/,10X,35HACCEL5 CORRECTED FROM LESTSQ ACCEL4/)          LES 104
 1010 FORMAT((1X,8(F8.3,F9.3)))                                          LES 105
 1020 FORMAT (1X,/,10X,38HINTEGRATED VELOCITY FROM LESTSQ ACCEL5/)       LES 106
 1040 FORMAT (1X,/,10X,48HCORRECTION TO ACCEL5 FROM LEASTSQUARING VELOCI LES 107
     TY/)                                                                LES 108
 1050 FORMAT (1X,/,10X,21HLEASTSQUARED VELOCITY/)                        LES 109
 1060 FORMAT (//,10X,12HDENOM = ZERO/)                                   LES 110
 1070 FORMAT (1X,I6,2F12.4)                                              LES 111
 1080 FORMAT (1X,/,10X,29HLEAST SQUARING ROUTINE -- N =,I5//)            LES 112
 1090 FORMAT (1X,2F10.3,I5,2F10.3)                                       LES 113
 2000 FORMAT (5X,4HTIME,6X,3HVEL)                                        LES 114
 2010 FORMAT (5X,4HTIME,4X,5HACCEL)                                      LES 115
 2020 FORMAT (5X,4HTIME,4X,5HACCEL)                                      LES 116
 2030 FORMAT (1X,/,6X,3HDIS,7X,4HAREA,4X,1HN,5X,5HTLAST,3X,9HTIME(N+1))  LES 117
 2040 FORMAT (4X,5HNPASS,7X,1HA,12X,1HB)                                 LES 118
      RETURN                                                             LES 119
      END                                                                LES 120


      SUBROUTINE INTERP (DELT,N2,NPASS,ENTER,EXIT,NEWWAY)                INT 001
      IMPLICIT REAL*4 (A-H,O-Z) \
C                                                                        INT 002
C      DATA TO BE INTERPOLATED IS ENTERED TO SUBROUTINE VIA ARRAY ENTER. INT 003
C      DURING INTERPOLATION, IT IS TEMPORARILY STORED IN THE SECOND      INT 004
C      ARRAY OF COMMON AND MUST BE TRANSFERRED BACK TO THE PROPER ARRAY  INT 005
C      AFTER IT LEAVES THE SUBROUTINE.                                   INT 006
C                                                                        INT 007
      COMMON ACCEL(5500),TIME(5500),TITLE(20),VEL(3000),DISP(3000),LAST
```

```
      DIMENSION ENTER(300), EXIT(3000)
C
      N = N2*10 + 1
      WRITE (3,1020) N
C
      LL=1
      NN = N2 + 1
      ATIME = 0.0
      DO 30 I=2,NN
      DIFF = (ENTER(I)-ENTER(I-1))/0.2
      TIMEI = (I-1)*0.2
      TIMEM1 = TIMEI-0.2
      DO 20 L=LL,N
      IF (ATIME .GE. TIMEI) GO TO 30
      IF(NEWWAY .EQ. 1) GO TO 10
      IF (NPASS .EQ. 1 .AND. TIMEI .GT. ATIME) ACCEL(L)=ACCEL(L)-DIFF
   10 EXIT(L)=ENTER(I-1)+(ENTER(I)-ENTER(I-1))*(ATIME-TIMEM1)/(TIMEI -
     1 TIMEM1)
   20 ATIME = L*DELT
   30 LL=L
      EXIT(N)=ENTER(NN)
      IF(NEWWAY .EQ. 1) GO TO 40
      ACCEL(N) = ACCEL(N) - DIFF
      LN = N - 8
   40 WRITE (3,1000)
      WRITE (3,1010) (EXIT(I), I=1,8)
      WRITE (3,1010) (EXIT(I), I=LN,N)
C
 1000 FORMAT (1X,/,10X,38HINTERPOLATED DATA STORED IN TIME ARRAY/)
 1010 FORMAT (16(F8.3))
 1020 FORMAT (1X,/,10X,26HINTERPOLATING ROUTINE - N=,I5//)
      RETURN
      END
```

```
      SUBROUTINE HOLWAY(NO2)
      IMPLICIT REAL*4 (A-H,O-Z)
C
C     HOLLOWAY RUNNING MEAN FILTER
C
C      THIS SUBROUTINE TAKES THE AVERAGE OF 19 POINTS AND REPLACES THE
C      TENTH (OR MIDPOINT OF THE WINDOW WIDTH) POINT WITH THAT VALUE
C
      COMMON ACCEL2(5500),ATEMP(5500),TITLE(20),ACCEL5(3000),D(3000),
     1LAST
C
C     WW = WINDOW WIDTH
C
      WW=19.
      N = NO2
      WRITE (3,1020) N
C
C      THE ATEMP ARRAY IS USED HERE AS A TEMPORARY STORAGE ARRAY
C
      J = (WW+1.0)/2.0
      IF(J .GT. N) J = N
      WW = 2.0*FLOAT(J) - 1.0
      DO 50 LL=1,N
      ATEMP(LL)=0.0
      I = 0
      DO 20 MM=1,J
      KK=LL+MM-1
      IF (KK .LE. N) GO TO 20
   10 I=I+1
      KK=N-I
   20 ATEMP(LL)=ATEMP(LL)+ACCEL2(KK)
```

```
      I = 1                                                             HOL 030
      DO 40 MM=2,J                                                      HOL 031
      KK=LL-MM+1                                                        HOL 032
      IF (KK .GT. 0) GO TO 40
   30 I=I+1                                                             HOL 034
      KK=I                                                              HOL 035
   40 ATEMP(LL)=ATEMP(LL)+ACCEL2(KK)                                    HOL 036
   50 ATEMP(LL)=ATEMP(LL)/WW                                            HOL 037
      WRITE (3,1000)                                                    HOL 038
      WRITE (3,1010) (ATEMP(LL),LL=1,N)                                 HOL 039
C                                                                       HOL 040
C      REPLACE FILTERED DATA INTO ACCEL ARRAY                           HOL 041
C                                                                       HOL 042
      DO 60 I=1,N                                                       HOL 043
      ACCEL2(I)=ATEMP(I)                                                HOL 044
   60 ATEMP(I)=(I-1)*0.02                                               HOL 045
C                                                                       HOL 046
 1000 FORMAT (1X,/,10X,25HACCEL FROM HOLOWAY FILTER/)                   HOL 047
 1010 FORMAT (1X,13F10.3)                                               HOL 048
 1020 FORMAT (1X,/,10X,34HHOLLOWAY RUNNING MEAN FILTER - N =,I5//)      HOL 049
      RETURN                                                            HOL 050
      END                                                               HOL 051


      SUBROUTINE ORMSBY (ISYM,FSUBC,FSUBT,DELT,AENTER,NSHORT)           ORM 001
      IMPLICIT REAL*4 (A-H,O-Z)
C                                                                       ORM 002
C     ORMSBY FILTER                                                     ORM 003
C                                                                       ORM 004
C      SUBROUTINE ORMSBY ACTS AS A LOW PASS FILTER FILTERING OUT ALL    ORM 005
C      FREQUENCIES GREATER THAN "FSUBT" CPS.  A LOW PASS FILTER ALLOWS  ORM 006
C      LOW FREQUENCIES TO PASS THROUGH WHILE FILTERING OUT THE HIGHER   ORM 007
C      FREQUENCIES DUE TO ACCELOROMETER AND DIGITIZATION ERRORS         ORM 008
C      UNFILTERED DATA IS CONVEYED VIA AENTER AS A PARAMETER OF ORMSBY, ORM 009
C      FILTERED DATA RETURNS VIA ATEMP ARRAY IN COMMON                  ORM 010
C                                                                       ORM 011
     COMMON ACCEL(5500),ATEMP(5500),TITLE(20),VEL(3000),DISP(3000),LAST
      COMMON /ORMS/ H(275),NN                                           ORM 013
      DIMENSION AENTER(5500)                                            ORM 014
C                                                                       ORM 015
C      N IS THE NUMBER OF POINTS IN ACCELOGRAM TO BE FILTERED.          ORM 016
C      NN IS THE NUMBER OF FILTER WEIGHTS IN EACH HALF OF FILTER.       ORM 017
C      2*NN+1 IS THE TOTAL NUMBER OF FILTER WEIGHTS.                    ORM 018
C                                                                       ORM 019
      WRITE (3,1000) ISYM                                               ORM 020
      N = (FLOAT(LAST+2)/100.0)/DELT                                    ORM 021
      IF (DELT .EQ. 0.2) N=N+1                                          ORM 022
      ALS = ABS(FSUBC-FSUBT)                                            ORM 023
      WRITE (3,1020) FSUBC,FSUBT,ALS,DELT                               ORM 024
      ALR = DELT*(ABS(FSUBC-FSUBT))                                     ORM 025
      NN=1.0/ALR                                                        ORM 026
      NNP1 = NN+1                                                       ORM 027
      IF (NN .GT.N) NN=N                                                ORM 028
C                                                                       ORM 029
      WRITE (3,1070) N                                                  ORM 030
      WRITE (3,1090) NN                                                 ORM 031
C                                                                       ORM 032
      PI=3.1415926535                                                   ORM 033
      ALC=FSUBC*DELT                                                    ORM 034
      ALT=ALC+ALR                                                       ORM 035
```

```
                                                                        ORM 036
C                                                                       ORM 037
C       THE FILTER WEIGHTS ARE CALCULATED                               ORM 038
C                                                                       ORM 039
      DO 10 MN=2,NN                                                     ORM 040
      MM=MN-1                                                           ORM 041
      H(MN)=(COS(2.*PI*ALC*MM)-COS(2.*PI*ALT*MM))/(2.*ALR*(PI*MM)**2)   ORM 042
   10 SUM = SUM+H(MN)                                                   ORM 043
C                                                                       OPM 044
C       H(1) IS THE VALUE OF THE FILTER WEIGHT AT THE CENTER            ORM 045
C                                                                       ORM 046
      H(1)=ALT+ALC                                                      ORM 047
      SUM=2.0*SUM+H(1)                                                  ORM 048
C                                                                       ORM 049
C       IF NN IS LESS THAN N, NOT ALL THE WEIGHTS ARE USED AND AN       ORM 050
C       ADJUSTMENT MUST BE MADE SO THAT THE SUM OF THE WEIGHTS IS ONE.  ORM 051
C                                                                       ORM 052
      XS = 0.0                                                          ORM 053
      DO 20 I=1,NN                                                      ORM 054
      H(I)=H(I)/SUM                                                     ORM 055
   20 XS=XS+H(I)                                                        ORM 056
      WRITE (3,1030)                                                    ORM 057
      WRITE (3,1040) (H(I),I=1,NN)                                      ORM 058
C                                                                       ORM 059
C       THE FILTER WEIGHTS ARE APPLIED TO THE EQUALLY SPACED DATA       ORM 060
C       AND STORED IN THE ATEMP ARRAY.                                  ORM 061
C                                                                       ORM 062
   30 DO 110 LL=1,N                                                     ORM 063
      ATEMP(LL)=0.0                                                     ORM 064
      I=0                                                               ORM 065
      K=0                                                               ORM 066
C                                                                       ORM 067
C       LOOP 60 APPLIES RIGHT PORTION OF FILTER WEIGHTS TO ACCELEROGRAM. ORM 068
C                                                                       ORM 069
      DO 60 MM=1,NN                                                     ORM 070
      KK=LL+K                                                           ORM 071
      IF (KK .LE. N) GO TO 50
   40 IF (ISYM .EQ. 0) GO TO 70                                         ORM 073
      I=I+1                                                             ORM 074
      KK=N-I                                                            ORM 075
      IF (KK .LE. 0) GO TO 60                                           ORM 076
   50 ATEMP(LL)=ATEMP(LL)+H(MM)*AENTER(KK)                              ORM 077
   60 K=K+1                                                             ORM 078
   70 K=1                                                               ORM 079
      I=1                                                               ORM 080
C                                                                       ORM 081
C       LOOP 100 APPLIES LEFT PORTION OF FILTER WEIGHTS TO ACCELEROGRAM. ORM 082
C                                                                       ORM 083
      DO 100 MM=2,NN                                                    ORM 084
      KK=LL-K                                                           ORM 085
      IF (KK .GT. 0) GO TO 90
   80 IF (ISYM .EQ. 0) GO TO 110                                        ORM 087
      I=I+1                                                             ORM 088
      KK=I                                                              ORM 089
      IF (KK .GT. N) GO TO 110                                          ORM 090
   90 ATEMP(LL)=ATEMP(LL)+H(MM)*AENTER(KK)                              ORM 091
  100 K=K+1                                                             ORM 092
  110 CONTINUE                                                          ORM 093
      SUM = 2.0*XS-H(1)                                                 ORM 094
      IF (ABS(SUM-1.0) .GT. 0.001) WRITE (3,1080) SUM                   ORM 095
      IF (NSHORT .EQ. 1) WRITE (3,1048)                                 ORM 096
      IF (NSHORT .EQ. 0) WRITE (3,1050)                                 ORM 097
      WRITE (3,1060) (ATEMP(I),I=1,N)                                   ORM 098
C                                                                       ORM 099
 1000 FORMAT (1X,/,2X,8H ISYM = ,I5,/)                                  ORM 100
 1020 FORMAT (10X,/,2X,8HFSUBC = ,F10.7,/,                              ORM 101
     .              2X,8HFSUBT = ,F10.7,/,                              ORM 102
     .              2X,8H  ALS = ,F10.7,/,                              ORM 103
     .              2X,8H DELT = ,F10.7,/)                              ORM 104
```

```
1030 FORMAT (1X,/,10X,21HORMSBY FILTER WEIGHTS/)
1040 FORMAT (1X,8F10.6)
1048 FORMAT (1X,/,10X,30HORMSBY FILTERED DATA AT .1 SEC/)
1050 FORMAT (1X,/,10X,30HORMSBY FILTERED DATA AT .2 SEC/)
1060 FORMAT((1X,4(2F10.3)))
1070 FORMAT(1X,/,10X,27HORMSBY FILTER ROUTINE - N =,I5//)
1080 FORMAT (1X,//,10X,24HSUM OF ORMSBY WEIGHTS IS,E12.5//)
1090 FORMAT (/,10X,3HTHE,I5,48HFILTER WEIGHTS CENTER AND RIGHT OF CENTE
    .R FOLLOW/)
     RETURN
     END



     SUBROUTINE HORIZ (NPASS)
     IMPLICIT REAL*4 (A-H,O-Z)
C    THIS SUBROUTINE REPLACES THE CURRENT HORIZONTAL BASELINE WITH A
C    NEW HORIZONTAL BASELINE SUCH THAT THE AREA UNDER THE ACCELERATION
C    CURVE IS ZERO.
C
     COMMON ACCEL(5500),TIME(5500),TITLE(20),RET(3000),TEMP(3000),LAST
C
C
C
     IF(NPASS .EQ. 2) GO TO 50
     IF (NPASS .EQ. 5) GO TO 70
     IF (NPASS .EQ. 6) GO TO 90
C
     N = LAST/2
     DT = 0.02
C
     AREA = 0.0
C
     DO 10 I=1,N
  10 AREA = AREA + (TEMP(I) + TEMP(I+1))*DT/2.0
C
     NP1 = N + 1
     CORREC = AREA/TIME(NP1)
  20 WRITE (3,1020) AREA
     WRITE (3,1030) CORREC
C
     DO 30 I=1,NP1
     RET(I) = ACCEL(I) - CORREC
  30 ACCEL(I) = RET(I)
     LLAST = LAST - 8
     WRITE (3,1060)
     WRITE (3,1040)
     WRITE (3,1010) (TIME(I), RET(I), I=1,8)
     WRITE (3,1010) (TIME(I), RET(I), I=LLAST,LAST)
C
     IF (NPASS .EQ. 5) GO TO 68
C
C    INTEGRATE TO GET VELOCITY
C
     TEMP(1) = 0.0
C
     DO 40 I=1,N
     DT = TIME(I+1) - TIME(I)
  40 TEMP(I+1) = TEMP(I) + (RET(I) + RET(I+1))*DT*0.5
     WRITE (3,1070)
     WRITE (3,1050)
     WRITE (3,1010) (TIME(I), TEMP(I), I=1,8)
     WRITE (3,1010) (TIME(I), TEMP(I), I=LLAST,LAST)
C
     RETURN
  50 CONTINUE
     N = LAST/2
```

```
      RET(1) = 0.0
      DO 60 I=1,N
      DT = TIME(I+1) - TIME(I)
   60 RET(I+1) = RET(I) + (ACCEL(I) + ACCEL(I+1))*DT*0.5
      WRITE (3,1070)
      WRITE (3,1050)
      WRITE (3,1010) (TIME(I), RET(I), I=1,8)
      WRITE (3,1010) (TIME(I), RET(I), I=LLAST,LAST)
C
   68 RETURN
   70 N = LAST
      DT = 0.01
      AREA = 0.0
      DO 80 I=1,N
   80 AREA = AREA + (ACCEL(I) + ACCEL(I+1))*DT/2.0
      CORREC = AREA/TIME(N)
      GO TO 20
   90 N = LAST/2
      DT = 0.02
      AREA = 0.0
      DO 100 I=1,N
  100 AREA = AREA + (ACCEL(I) + ACCEL(I+1))*DT/2.0
      CORREC = AREA/TIME(N)
      GO TO 20
C
 1010 FORMAT((1X,8(F8.3,F9.3)))
 1020 FORMAT (1X,/,10X,9H  AREA = ,F10.6)
 1030 FORMAT (10X,9HCORREC = ,F10.6)
 1040 FORMAT (5X,4HTIME,4X,5HACCEL)
 1050 FORMAT (5X,4HTIME,6X,3HVEL)
 1060 FORMAT (1X,/,10X,42HACCEL ADJUSTED FOR ZERO VELOCITY VIA HORIZ/)
 1070 FORMAT (1X,/,10X,29HINTEGRATED VELOCITY VIA HORIZ/)
      END
```

```
      SUBROUTINE PLTDAT (NPLOT,DELT,XRED,YRED,NO2)
      IMPLICIT REAL*4 (A-H,O-Z)
C
C     THIS SUBROUTINE PLOTS THE FILTERED GROUND ACCELERATION,
C     VELOCITY AND DISPLACEMENT FOR THE DIGITIZED SEISMIC DATA.
C     IN ADDITION, THE ORMSBY LOW-PASS FILTER IS SHOWN.
C
      COMMON ACCEL8(5500),TIME(5500),TITLE(20),VEL4(3000),DISP(3000),
     1LAST
      COMMON/PLTBUF/IBUF(1000)
      COMMON /ORMS/ WTS(275),NN
C
      N = NO2 - 1
      WRITE (3,4000) N
C
      LAST = N
      DO 6 I=1,N
    6 TIME(I)=(I-1)*DELT
C
      SIZE = 3.0
      XLEN = 12.0
C
      CALL PLOTS(IBUF,1000,5)
      IF (XRED .EQ. 1.0 .AND. YRED .EQ. 1.0) GO TO 8
      CALL SETFACT (XRED,YRED)
    8 IF (NPLOT .EQ. 4) GO TO 55
      CALL PLOT (0.,-11.,-3)
   11 CALL SCALE (TIME,XLEN,LAST,1)
      CALL SCALE (ACCEL8,SIZE,LAST,1)
C
C     PLOT ACCELERATION
C
```

```
      COR = 10.5-SIZE
      CALL PLOT (0.0,COR,-3)
      CALL AXIS (0.0,3.0," ",1,XLEN,0.,TIME(LAST+1),TIME(LAST+2))
      CALL AXIS (0.0,0.0,"GRD ACCEL IN G" ,14,SIZE,90.,ACCEL8(LAST+1),
     1 ACCEL8(LAST+2))
      CALL LINE (TIME,ACCEL8,LAST,1,0,0)
      BSLINE = ABS(ACCEL8(LAST+1)/ACCEL8(LAST+2))
      CALL PLOT (XLEN,8SLINE,3)
      CALL PLOT (0.0,BSLINE,2)
      CALL SYMBOL (0.0,-0.5,0.25,TITLE,0.,80)
      CALL PLOT (0.,-9.0,-3)
      IF (NPLOT .EQ. 1) GO TO 999
C
C     PLOT VELOCITY
C
      SIZE = (9.0-SIZE)/2.0
      COR = SIZE + 1.0
      CALL PLOT (0.,COR,-3)
      CALL SCALE (VEL4,SIZE,LAST,1)
      DIVY = 10.0
      CALL AXIS (0.0,0.0,"VEL. IN CM/SEC." ,15,SIZE,90.,VEL4(LAST+1),
     1 VEL4,LAST+2))
      CALL LINE (TIME,VEL4,LAST,1,0,0)
      8SLINE = ABS(VEL4(LAST+1)/VEL4(LAST+2))
      CALL PLOT (XLEN,8SLINE,3)
      CALL PLOT (0.0,BSLINE,2)
      CALL PLOT (0.,-5.0,-3)
C
C     PLOT DISPLACEMENT
C
      CALL PLOT (0.,0.5,-3)
      CALL SCALE (DISP,SIZE,LAST,1)
      CALL AXIS (0.0,0.0,"TIME IN SECONDS" ,-15,XLEN,0.,TIME(LAST+1),
     1 TIME(LAST+2))
      DIVY = 10.0
      CALL AXIS (0.0,0.0,"DISP. IN CM." ,12,SIZE,90.,DISP(LAST+1),DISP(L
     1AST+2))
      CALL LINE (TIME, DISP,LAST,1,0,0)
      8SLINE = ABS(DISP(LAST+1)/DISP(LAST+2))
      CALL PLOT (XLEN,BSLINE,3)
      CALL PLOT (0.0,8SLINE,2)
      CALL PLOT (0.,-0.5,-3)
      IF (NPLOT .EQ. 2) GO TO 999
C
C     PLOT ACCEL5
C
   55 START5 = XLEN + 2.0
      CALL PLOT (START5,5.0,-3)
      CALL SCALE (TIME,XLEN,NN,1)
      CALL AXIS (0.0,0.0,"TIME IN SECONDS" ,-15,XLEN,0.,TIME(NN+1),
     1 TIME(NN+2))
      CALL SCALE (WTS,5.,NN,1)
      CALL AXIS (0.0,0.0,"FILTER WEIGHTS" ,14,5.,90.,WTS(NN+1),WTS(NN+2)
     1)
      CALL LINE (TIME,WTS,NN,1,0,0)
      CALL SYMBOL (2.0,5.0,0.25,TITLE,0.,80)
      CALL SYMBOL (2.0,-2.0,0.25,"ORMSBY FILTER" ,0.0,13)
  999 CONTINUE
      CALL PLOT(20.0,0.0,999)
C
 4000 FORMAT (1X,/,10X,14HPLOT DATA  N =,I6)
      RETURN
      END
```

To compile the program for the CDC 6400 machine, the unlabelled statements (those not containing a name and number in columns 73 through 80) should be removed and the following statements inserted in their sequence locations.

```
      PROGRAM DAISMA(INPUT=129,OUTPUT=129,PLOT,PUNCH,TAPE1=INPUT,       DAI 001
     .TAPE3=OUTPUT,TAPE5=PLOT,TAPE7)                                    DAI 002

      COMMON ACCEL(5500),TIME(5500),TITLE(8),VEL(3000),DISP(3000),LAST  DAI 036

      COMMON ACCEL(5500),TIME(5500),TITLE(8),VEL(3000),DISP(3000),LAST  PH1 008
   10 READ (1,2050) (TITLE(I), I=1,8)                                   PH1 014
      IF (EOF(1)) 20,30                                                 PH1 015
      CALL REMARK (31HNORMAL TERMINATION, END OF DATA)                  PH1 017
   30 WRITE (3,2060) (TITLE(I), I=1,8)                                  PH1 019
      IF (NPUNCH .EQ. 1 .OR. NPUNCH .EQ. 2) PUNCH 2050, (TITLE(I),I=1,8) PH1 057
      IF (NT .GE. 2998) 60,70                                           PH1 071
   70 READ (7,2040) TIME(NT), ACCEL(NT)                                 PH1 074
      IF(EOF(7)) 80,50                                                  PH1 075
      IF (NT .GE. 749) 90,100                                           PH1 082
  100 READ (1,1000) (TIME(NT*4+I),ACCEL(NT*4+I), I=1,4)                 PH1 086
      IF(EOF(1)) 110,85                                                 PH1 087
      READ (1,1000) (X(NX*4+I), Y(NX*4+I), I=1,4)                       PH1 123
      IF(EOF(1)) 190,180                                                PH1 124
  380 CALL INITIAL (0,5,0.3,0,0)                                        PH1 228
      CALL AXIS (0.,0.0,19HREL. TIME IN COUNTS,-19,XLEN,0.,XMIN,SCALX,  PH1 234
     .20.0)                                                             PH1 235
      CALL AXIS (0.,0.0,21HREL. ACCEL. IN COUNTS,21,SIZE,90.,YMIN,SCALY, PH1 236
     .10.)                                                              PH1 237
      IF (LAST .GT. 200) 400,410                                        PH1 239
  440 CALL ENDPLT                                                       PH1 262
 2050 FORMAT ( 8A10)                                                    PH1 287
 2060 FORMAT (1H,8A10,//)                                               PH1 288
```

```
      SUBROUTINE TRKO10
C
C     *****************************************************************
C
C     THIS SUBROUTINE READS DATA FROM AN ELECTRAK DATA TAPE AND OUTPUTS
C     ON TAPE 7 DATA WHICH IS IN A FORM ACCESSIBLE BY PHASE1 PROGRAM.
C     THIS PROGRAM IS A PATTERN MATCH PROCEDURE, PROBABLY
C     BETTER SUITED TO WRITING IN SNOBOL4.  THE TARGET PATTERN IS
C     ZZZZZZZNNNNSNNNNNSNNNNNCCCCCCCCCCCCCCCCCCCZZZZ...
C            ^          ^                   ^
C     WHERE  Z IS ANY CHARACTER
C            N IS ANY DECIMAL DIGIT
C            S IS A SIGN (+ OR -, <- READS AS A ;><+  READS AS A ]> )
C            C IS A SET OF CHARACTERS ASSOCIATED WITH THE NUMBER
C     THE GOAL IS TO OUTPUT RECORDS OF THE ENTIRE 36 CHARACTER
C     RECORD AND THE COMMENTS BETWEEN THIS AND THE PRECEEDING
C     PATTERN.  THESE RECORDS ARE OUTPUT ON TAPE7.
C
C     *****************************************************************
C
      IMPLICIT INTEGER (A-Z)
      DIMENSION BUFS(1000)
      DIMENSION DAT(8)
      EQUIVALENCE(DAT(1),WD1),(DAT(2),WD2),(DAT(3),CWD1),(DAT(4),CWD2)
      FGET(WD,N,V)=(WD.A..N.SHIFT(77B,60-N)).OR.SHIFT(V,60-N)
      DATA POS,LSTPOS/2*0/, TARG1,TARG2,I,RC/1R],1R;,1,0/
C
      REWIND 7
   10 BUFFERIN (1,0) (BUFS(I),BUFS(1000))
      IF (UNIT(1)) 100,20,90
   20 WRITE (3,21)
   21 FORMAT (1X,/,10X,*END OF TAPE*,//)
      RETURN
   90 CALL REMARK (20HTRAK1, PARITY ERROR )
  100 CALL LENGTHX(1,SIZE,WASTE)
      RC=RC+1
      IF (WASTE .NE. 0) PRINT 110, WASTE
  110 FORMAT (*  $$UNUSED BITS IN LAST READ:*I4)
      LIMET=SIZE+I-1
      J=1
  150 WD=BUFS(J)
C
C     THIS LOOP SEEKS A PAIR OF SIGNS (+,-) SEPARATED BY 5 CHARACTERS
C     SINCE ELECTRAK - SIGNS READ AS ';' THEY ARE REPLACED BY '-'
C     AND + SIGNS READ AS ']' THEY ARE REPLACED BY THE '+'.
C     THIS CONVERSION IS GLOBAL, I.E. IT IS DONE WHETHER OR NOT
C     THE PATTERN SEARCH SUCCEEDS (N POINTS TO THE FIRST SIGN).
C
      DO 350 N=6,60,6
      CHAR=SHIFT(WD,N).AND.77B
      IF (CHAR .NE. TARG1) GO TO 260
      BUFS(J)=FGET(BUFS(J),N,45B)
      GO TO 270
  260 IF (CHAR .NE. TARG2) GO TO 350
      BUFS(J)=FGET(BUFS(J),N,46B)
  270 JJ=J  $  NN=N+36
      IF (NN .LE. 60) GO TO 280
      NN=NN-60
      JJ=JJ+1
  280 CHAR2=SHIFT(BUFS(JJ),NN).AND.77B
      IF (CHAR2 .NE. TARG1) GO TO 340
      BUFS(JJ)=FGET(BUFS(JJ),NN,45B)
      GO TO 450
  340 IF (CHAR2 .EQ. TARG2) GO TO 400
  350 CONTINUE
C
C     PATTERN MATCH HAS FAILED
```

```
C
      GO TO 1000
C
C     PATTERN MATCH HAS SUCCEEDED
C
  400 BUFS(JJ)=FGET(BUFS(JJ),NN,46B)
  450 WD=BUFS(J)
C
C     THE COMPUTED GOTO BRANCHES TO THE APPROPRIATE MASK CODE
C     FOR THE POSITION OF THE FIRST SIGN.  THIS CURSER LOCATION
C     CONTINUES ALL THE SHIFT-MASK OPERATIONS.
C
      GO TO (470,470,470,470,480,480,485,490,490,500),N/6
  470 WD1=SHIFT(BUFS(J-1).A..N.MASK(30+N),18+N).OR.SHIFT(WD
     1.AND.MASK(N+18),N+18)
      WD2=SHIFT(BUFS(J+1).AND.MASK(N+6),N+6).OR.(SHIFT(WD.A..N.MASK
     1(N+18),6+N))
      GO TO 520
  480 WD1=SHIFT(WD,N+18).A.77777777777777778
      WD2=SHIFT(WD.A..N.MASK(N+18),N+6)
     1.OR.(SHIFT(BUFS(J+1),N+6).AND..N.MASK(54-N))
      GO TO 520
  485 WD1=WD.AND.77777777777777778
      WD2=SHIFT(BUFS(J+1),48).AND.77777777777777778
      GO TO 520
  490 WD1=SHIFT(WD.A..N.MASK(N-30),N-42).OR.SHIFT(BUFS(J+1)
     1.AND.MASK(N-42),N-42)
      WD2=SHIFT(BUFS(J+1),N+6).AND.77777777777777778
      GO TO 520
  500 WD1=SHIFT(WD.AND..N.MASK(30),18).OR.SHIFT(BUFS(J+1)
     1.AND.MASK(18),18)
      WD2=SHIFT(BUFS(J+1).A..N.MASK(18),6).OR.(SHIFT(BUFS(J+2),6)
     1.AND.77B)
C
C     THIS IS A RELATIVELY RELIABLE WAY TO TEST FOR THE TARGET
C     PATTERN.  THE PREVIOUS STATEMENTS PUT THE 16 DIGITS OF THE
C     TARGET INTO 2 WORDS OF MEMORY, WD1 AND WD2.  THESE WORDS ARE
C     IN 8RZZZZZZZZ FORMAT.
C
  520 DO 540 M=18,60,6
      TEST=(SHIFT(WD1,M).AND.77B)-33B
      IF (TEST .LT. 0 .OR. TEST .GT. 13B) GO TO 800
      TEST=(SHIFT(WD2,M).AND.77B)-33B
      IF (TEST .LT. 0 .OR. TEST .GT. 13B) GO TO 800
  540 CONTINUE
      WRITE (7,700) (DAT(JB),JB=1,2)
  700 FORMAT (2R8)
      WRITE (3,701)(DAT(I), I=1,2)
  701 FORMAT (1X,2R8)
      J=JJ+1
      IF (J .GT. LIMET-4) 1010,150
  800 PRINT 801,BUFS(J-1),BUFS(J),BUFS(J+1),N,J,RC
  801 FORMAT(3(2X,O20),* BIT COUNT=*I2,*:  BUFFER LOC=*I3,
     1 *;  RECORD COUNT=*I4)
 1000 J=J+1
      IF (J .GT. LIMET-4) 1010,150
 1010 I=1
      IF (LIMET-JJ .GT. 15) JJ=LIMET-6
      DO 1050 N=JJ,LIMET
      BUFS(I)=BUFS(N)
      I=I+1
 1050 CONTINUE
      GO TO 10
      END
```

```
      COMMON ACCEL(5500),TIME(5500),TITLE(8),VEL(3000),DISP(3000),LAST   PH2 016
      READ (1,1090) (TITLE(I), I=1,8)                                    PH2 029
      WRITE (3,2000) (TITLE(I), I=1,8)                                   PH2 030
   78 IF (T .LE. 0.0 .OR. CD .LE. 0.0) 110,90                            PH2 153
 1090 FORMAT ( 8A10)                                                     PH2 490
 2000 FORMAT (1H,8A10,//)                                                PH2 491

      COMMON ACCEL(5500),TIME(5500),TITLE(8),VEL(3000),DISP(3000),LAST   RED 006
      READ(1,1000)(TIME(NT*4+I),ACCEL(NT*4+I),I=1,4)                     RED 012
      IF(EOF(1))20,10                                                    RED 013
      IF (LAST .GT. 2998) 30,40                                          RED 019
      IF (XN .GT. 54.95) 50,60                                           RED 024
      CALL REMARK (28HINPUT DATA OVERLOADS PROGRAM)                      RED 029

      COMMON ACCEL(5500),TIME(5500),TITLE(8),VEL(3000),DISP(3000),LAST   DAT 005

      COMMON ACCEL(5500),TEMP(5500),TITLE(8),VEL(3000),TIME(3000),LAST   EQL 008

      COMMON ACCEL(5500),TIME(5500),TITLE(8),VEL(3000),TEMP(3000),LAST   LES 006
      IF (TIME(I) .GE. 0.0) 20,10                                        LES 017
      IF (DENOM .EQ. 0.0) 50,60                                          LES 042

      COMMON ACCEL(5500),TIME(5500),TITLE(8),VEL(3000),DISP(3000),LAST   INT 008

      COMMON ACCEL2(5500),ATEMP(5500),TITLE(8),ACCEL5(3000),D(3000),LAST HOL 008
      IF (KK .GT. N) 10,20                                               HOL 026
      IF (KK .LE. 0) 30,40                                               HOL 033

      COMMON ACCEL(5500),ATEMP(5500),TITLE(8),VEL(3000),DISP(3000),LAST  ORM 012
      IF (KK .GT. N) 40,50                                               ORM 072
      IF (KK .LE. 0) 80,90                                               ORM 086

      COMMON ACCEL(5500),TIME(5500),TITLE(8),RET(3000),TEMP(3000),LAST   HOR 006
```

```
 COMMON ACCEL8(5500),TIME(5500),TITLE(8),VEL4(3000),DISP(3000),LAST PLT 007
 CALL INITIAL (0,5,0.3,0,0)                                         PLT 020

11 CALL SCALE (TIME,XLEN,LAST,1,20.0)                               PLT 025
 CALL SCALE (ACCEL8,SIZE,LAST,1,10.)                                PLT 026
 CALL AXIS (0.0,3.0,1H ,1,XLEN,0.,TIME(LAST+1),TIME(LAST+2),10.)    PLT 032
 CALL AXIS (0.0,0.0,14HGRD ACCEL IN G, 14,SIZE, 90.,ACCEL8(LAST+1), PLT 033
1 ACCEL8(LAST+2),10.0)                                              PLT 034
 CALL SCALE (VEL4,SIZE,LAST,1,10.0)                                 PLT 048
 CALL AXIS (0.0,0.0,15HVEL. IN CM/SEC., 15,SIZE, 90.,VEL4(LAST+1),  PLT 050
1 VEL4(LAST+2),DIVY)                                                PLT 051
 CALL SCALE (DISP,SIZE,LAST,1,10.)                                  PLT 061
 CALL AXIS (0.0,0.0,15HTIME IN SECONDS,-15,XLEN,0.,TIME(LAST+1),    PLT 062
1 TIME(LAST+2),10.0)                                                PLT 063
 CALL AXIS (0.0,0.0,12HDISP. IN CM., 12,SIZE, 90.,DISP(LAST+1),DISP PLT 065
1(LAST+2),DIVY)                                                     PLT 066
 CALL SCALE (TIME,XLEN,NN,1,10.)                                    PLT 078
 CALL AXIS (00.0,0.0,15HTIME IN SECONDS,-15,XLEN,0.,TIME(NN+1),     PLT 079
1 TIME(NN+2),10.0)                                                  PLT 080
 CALL SCALE (WTS,5.,NN,1,10.)                                       PLT 081
 CALL AXIS (0.0,0.0,14HFILTER WEIGHTS, 14,5., 90.,WTS(NN+1),WTS(NN+ PLT 082
12),10.0)                                                           PLT 083
 CALL SYMBOL (2.0,-2.0,0.25,13HORMSBY FILTER,0.0,13)                PLT 086
 CALL ENDPLT                                                        PLT 088
```

If library program IMSL is available on the CDC computer then the addition of the following statements will produce a printer plot similar to Figure 51-a.

```
 DATA PTITLE/160*" "/                                               EQL 010
 CALL USPLH (TEMP,ACCEL,M,1,1,1,TITLE,ITEMP,IER)                    EQL 039
 COMMON /DRAW/ ITEMP(5151),PTITLE(160)                              LES 008
 DATA PTITLE/160*" "/                                               LES 009
 CALL USPLH (TIME,RET,NP1,1,1,1,TITLE,ITEMP,IER)                    LES 065
 CALL USPLH (TIME,TEMP,N,1,1,1,TITLE,ITEMP,IER)                     LES 083
```

```
      CALL USPLH (TIME,RET,NP1,1,1,1,TITLE,ITEMP,IER)                   LES 094
      COMMON /DRAW/ ITEMP(5151),PTITLE(160)                             HOR 007
      DATA PTITLE/160*" "/                                              HOR 008
      CALL USPLH (TIME,RET,LAST,1,1,1,TITLE,ITEMP,IER)                  HOR 035
      CALL USPLH (TIME,TEMP,LAST,1,1,1,TITLE,ITEMP,IER)                 HOR 049
      CALL USPLH (TIME,RET,LAST,1,1,1,TITLE,ITEMP,IER)                  HOR 061

      DIMENSION ITEMP(5151),PTITLE(160)                                 PH1 011
      DATA PTITLE/160*" "/                                              PH1 012
      CALL USPLH (TIME,ACCEL, LAST, 1,10,1,PTITLE,ITEMP,IER)            PH1 134
      CALL USPLH (X,Y,  NEXTRC,1,1,1,PTITLE,ITEMP,IER)                  PH1 135
      CALL USPLH (TIME,ACCEL, LAST,1, 1,1,PTITLE,ITEMP,IER)             PH1 265
      COMMON /DRAW/ ITEMP(5151),PTITLE(160)                             PH2 026
      DATA PTITLE/160*" "/                                              PH2 027
      CALL USPLH (TIME,ACCEL,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 078
      CALL USPLH (TIME,ACCEL,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 138
      CALL USPLH (TIME,ACCEL, NO2,1,1,1,TITLE,ITEMP,IER)                PH2 149
      CALL USPLH (TIME,ACCEL,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 198
      CALL USPLH (TIME,ACCEL,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 213
      CALL USPLH (TIME,ACCEL,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 260
      CALL USPLH (TIME,ACCEL,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 277
      CALL USPLH (TIME,ACCEL,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 303
      CALL USPLH (TIME,VEL  ,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 309
      CALL USPLH (TIME,ACCEL,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 369
      CALL USPLH (TIME,VEL  ,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 375
      CALL USPLH (TIME,DISP ,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 394
      CALL USPLH (TIME,ACCEL,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 456
      CALL USPLH (TIME,VEL  ,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 458
  475 CALL USPLH (TIME,DISP ,  NO2,1,1,1,TITLE,ITEMP,IER)               PH2 459
      COMMON /DRAW/ ITEMP(5151),PTITLE(160)                             RED 007
      DATA PTITLE/160*" "/                                              RED 008
      CALL USPLH (TIME,ACCEL,  LAST,1,1,1,TITLE,ITEMP,IER)              RED 061
      COMMON /DRAW/ ITEMP(5151),PTITLE(160)                             DAT 006
      DATA PTITLE/160*" "/                                              DAT 007
      CALL USPLH (TIME,ACCEL,  LAST,1,1,1,TITLE,ITEMP,IER)              DAT 038
      COMMON /DRAW/ ITEMP(5151),PTITLE(160)                             EQL 009
```

```
//MIKLOF   JOB T-675--MIKLOF,'MIKLOF',
//  CLASS=A

//        EXEC  PROC=UT6FLG,PARM.FORT=NODECK

//FORT.SYSIN   DD *
```

SOURCE DECK

```
//GO.FT03F001 dd sysout

//GO.FT03F001 DD SYSOUT=A,DCB=(RECFM=VBA,BLKSIZE=2000)
//FT01F001 DD *
//FT05F001 DD SYSOUT=B,DCB=(FUNC=I)
//FT02F001  DD  SYSOUT=8
//FT06F001  DD  DSN=INPUT,UNIT=TAPE,DISP=OLD,LABEL=(2,BLP),
//  VOL=SER=INPUT,DCB=(RECFM=F,LRECL=5040,BLKSIZE=5040)
//FT06F002  DD  DSN=INPUT,UNIT=AFF=FT06F001,DISP=OLD,LABEL=(3,BLP),
//  VOL=SER=INPUT,DCB=(RECFM=F,LRECL=5040,BLKSIZE=5040)
//FT07F001 DD  DSN=&&DISK07,UNIT=SYSDA,DISP=(,DELETE),
//   SPACE=(CYL,5)

/*
```

DATA DECK

```
//
```

CONTROL STATEMENTS FOR COMPILING DAISMA ON CDC 6400 COMPUTER

```
JOBCARD-NAME, BN________,CM117000,T20.
FTN(B=PUNCHB,OPT=1)
789 (ALL IN COLUMN ONE)
```

OBJECT DECK

```
6789 (ALL IN COLUMN ONE)
```

CONTROL STATEMENTS FOR DATA INPUT CDC 6400

```
JOBCARD-NAME,BN________,CM112000,T70.
INPUT.
789 (ALL IN COLUMN ONE)
```

IF AN ISML LIBRARY PROGRAM IS AVAILABLE THEN USE THE FOLLOWING CONTROL STATEMENTS TO PRODUCE A PRINTER PLOT

```
JOBCARD-NAME,BN________,CM117000,T70.
ATTACH(IMSL)
LDSET(LIB=IMSL)
INPUT.
```

BINARY DECK

```
789 (ALL IN COLUMN ONE)
```

DATA DECK

```
6789 (ALL IN COLUMN ONE)
```

APPENDIX E

TRAK 010 SUBROUTINE

1.1 Introduction

TRAK 010, written by Steve Kutoroff of Watershed Research, is a specialized subroutine that reads and analyzes data digitized on an Electrak digitizing machine.* The program is specialized in that it is for use only on data that has been taken by this type of machine. The digitized data is recorded on a 7-track magnetic tape in coded form and must be transformed into a general format so that it can be used by the final stages of processing in programs PHASE1 and PHASE2 for correction and integration.

TRAK 010 is written in the Fortran IV programing language and utilizes the fastest commands possible to read and analyze the data, i.e., buffering, shifting, and masking. In general, the shift and mask functions are Fortran intrinsic functions similar to ABS and SQRT that return a single value. The function SHIFT (A1,A2) commands the sixty bits of word A1 to move left circular A2 number of bit positions. If A2 is negative, the shift will move right, drop A2 number of bits off the end, and replace the first A2 bit positions with that of the first bit position (the sign).

The function MASK (A1) produces a word of 60 bits with ones in the leftmost A1 bit positions, and zero filled to the right. Masking operations are used in conjunction with SHIFT functions in the TRAK 010 subroutine to analyze the characters of a computer word.

---

* Applicable only for CDC 6400 Computer.

1.2 Purpose

The purpose for such a program is twofold. Data points are recorded from the Electrak digitizing machine to the magnetic tape in a continuous string of characters, each record of 36 characters placed immediately following the previous record. If one or more of these characters is not recorded, this interrupts the continuity of the information on the tape and the garbled data becomes useless to the computer. Secondly, the CDC 6400 computer interprets a "]" for a "+" sign and a ";" for a "-" sign. These incomplete records (less than 36 characters) and character differences are frequent enough to require such a corrective subroutine.

1.3 Subroutine TRAK 010

Each record is output on a magnetic tape as a string of 36 continuous characters and appears as follows:

CCCCSXXXXXSYYYYYFIXED-ADDRESS---CICI

where the CCCC represents a counter of the number of characters remaining in a block, the S represents a sign ("+" or "-"), the XXXXX is five digits of the X coordinate followed by another sign, and YYYYY is the five digits of the Y coordinate. The characters that follow give certain information about the data point and are explained in the section under Electrak Digitizing. Since these last 20 characters are not necessary in defining the location of the data point, they are bypassed in the TRAK 010 subroutine.

The usable information of the total 36 character record is the first 16 characters, i.e., the counter value, the X coordinate and the Y coordinate. Basically, the TRAK 010 subroutine reads this continuous output of characters and searches for a pair of signs (+ or -) that are six characters apart. When

it finds this target pattern, it checks the characters four to the left of the first sign and five to the right of the second sign. If all of these characters (except of course the signs) are decimal numbers, they are output on Tape 7 as a valid data point. This process continues until an end-of-file (EOF) tape mark is encountered indicating a completely digitized accelerogram component.

1.4 How the Subprogram Works

Basically TRAK 010 has four major sections. These are:

1. Buffer in 1000 words (10,000 characters) into memory (in the array BUFS) and check the status of this operation.
2. Locate a target pattern (two signs six characters apart).
3. Change the format of this target pattern to prepare it for testing and output.
4. Test the target pattern and output valid data points.

Section 1

The BUFFER-IN statement reads the first 1000 characters into the BUFS array. The UNIT statement is a utility subprogram which is used to check the status of the previous BUFFER-IN operation. It returns: (A) a -1 if no end-of-file or parity error is encountered, (B) a 0 if an end-of-file is encountered, and (C) a +1 if a parity error is encountered. A parity error is an error of nonagreement between the bit located in the first track (of a seven track tape) and the number of ones in the remaining six tracks. Since the Electrak digitizer records in "even" parity, it places a one in the first track if there is an even number of ones in the remaining six tracks (or leaves the first track blank if there is an odd number of ones in the remaining six tracks). Parity

error detection gives a 50% chance of detecting an error in the recording of information on the tape.

If a -1 is returned by the UNIT function, the utility subprogram LENGTHX is called and this returns the number of 60 bit words read (SIZE) and the number of unused bits in the last word read (WASTE). This information is retained and used later in the TRAK 010 subroutine.

If a zero is returned by the UNIT function, an end-of-file has been encountered in the BUFFER-IN operation indicating the end of data on the tape and control reverts back to the main program PHASE1.

If a +1 is returned by UNIT indicating a parity error, a remark is placed in the day file and the program continues.

Section 2

This section scans the current word in the BUFS array character by character searching for a pair of signs six characters apart. This task is performed by the SHIFT function. In the statement "CHAR = SHIFT(WD,N).AND. 77B", the word "CHAR" is replaced by the right side of the "=" sign. That is, SHIFT(WD,N) moves the 60 bits of the word "WD" (the first 10 characters of data) left circular N number of bit positions. For example, if "WD" contained the characters 1A2B3C4+DE, the bits for "WD" would look like this:

Character	1	A	2	B	3	C	4
Binary NO.	34	01	35	02	36	03	37
Bit	011100	000001	011101	000010	011110	000011	011111

Character	+	D	E
Binary NO.	45	04	05
Bit	100101	000100	000101

A shift of "WD", N (N= 6 first time through the loop) bit positions left circular would look like this:

Character	A	2	B	3	C	4	+
Bit	000001	011101	000010	011110	000011	011111	100101
Character	D	E	1				
Bit	000100	000101	011100				

The "77B" is a binary number representing a 60 bit word and appears like this:

Binary NO.	00	00	00	00	00	00	00
Bit	000000	000000	000000	000000	000000	000000	000000
Binary NO.	00	00	77				
Bit	000000	000000	111111				

The ".AND." performs the logical bit by bit product of the two words SHIFT(WD,N) and 77B. The final result (i.e., contents of "CHAR") appear like this:

Binary NO.	00	00	00	00	00	00	00
Bit	000000	000000	000000	000000	000000	000000	000000
Binary NO.	00	00	34				
Bit	000000	000000	011100				

"CHAR" is now a one character word with the contents 34B. This binary number represents the character "1". In the statement "IF(CHAR .NE. TARG1) GO TO 260", the contents of the word "CHAR" is compared to the contents of the word "TARG1" (where TARG1 represents a "+" sign). Since CHAR is not equal to TARG1 in our example, control proceeds to statement #260 where it is compared to "TARG2". "TARG2" is a word that represents a "-" sign. Again, since in our example "CHAR" is not a "-" sign, control reverts back to

the beginning of the loop and the second character of "WD" (i.e., the "A") is tested to see if it is a "+" or a "-" sign. This process continues until a sign is found.

As soon as a sign is found, the statement function "FGET" is called. A statement function is a user-defined, single-statement computation which performs a specific computation whenever it is referenced. In the TRAK 010 subroutine, FGET utilizes logical arithmetic and the SHIFT function to change the character "]" to "+" or ";" to "-". (Recall this is necessary because the characters "+" and "-" recorded by the Electrak digitizer is read as a "]" and ";" by the CDC 6400 computer.)

This character change operation is performed in the following manner. In the statement function FGET(WD,N,V) = (WD.A..N.SHIFT(77B,60-N)) .OR. SHIFT(V,60-N) the dummy arguments WD,N, and V are replaced by the actual arguments in the referencing statement. For example, the referencing statement "BUFS(J) = FGET(BUFS(J),N,45B)" WD of the statement function is substituted for BUFS(J), N is substituted for N, and V is substituted for 45B.

To illustrate this procedure, assume that the word "CHAR" contains the character "]". This is the Electrak's symbol for a "+" sign and must be changed to such before it can be output as usable data. "CHAR" is then compared to TARG1 ("]") and these two words are found to be equal. Therefore, the statement function "FGET" is referenced.

The word "WD" is replaced by the word BUFS(J) and appears like this:

1	A	2	B	3	C	4
011100	000001	011101	000010	011110	000011	011111
]	D	E				
110010	000100	000101				

77B is represented in bit form as:

000000 000000 000000 000000 000000 000000 000000
000000 000000 111111

SHIFT(77B,60-N) shifts the 77B twelve bit positions to the left

.A. (since N is now equal to 48, i.e., from loop #350, N = character location x 6 or N = 8 x 6 = 48) and appears as:

000000 000000 000000 000000 000000 000000 000000
111111 000000 000000

.N. is a logical operator which simply changes each bit in a word to what it is not, thus .N.SHIFT(77B,60-N) yields:

111111 111111 111111 111111 111111 111111 111111
000000 111111 111111

The .A. performs the logical product of the two 60 bit words BUFS(J) and .N.SHIFT(77B,60-N) which gives:

011100 000001 011101 000010 011110 000011 011111
000000 000100 000101

The word "V" (which has been replaced by 45B) looks like this:

.OR.

000000 000000 000000 000000 000000 000000 000000
000000 000000 100101

SHIFT(V,60-N) shifts V twelve bit positions left which yields:

000000 000000 000000 000000 000000 000000 000000
100101 000000 000000

.OR. is a logical operator which yields a "1" in that bit position if either word contains a "1" in the corresponding bit position.

thus: 011100 000001 011101 000010 011110 000011 011111
100101 000100 000101

Thus the word returned by FGET is a duplicate of the previous word BUFS(J) except that the eighth character has been changed from "]" to "+".

Since a sign was found in the eighth character of the first word BUFS(J), N = 48 (i.e., 6x8 = 48) and J = 1, the variables "JJ" and "NN" are set equal to "J" (i.e., 1) and N + 36 (i.e., 48 + 36 = 84). Because NN is greater than 60, JJ is set to JJ + 1 (i.e., 2) and NN is set to NN-60 or 24. This step prepares the subroutine to search in the second word of BUFS (the next 10 characters) for another sign (+ or -). Statement #280 "CHAR2 = SHIFT(BUFS(JJ),N).AND. 77B" is similar to the first statement in the loop where it compares the fourth character of the second word (BUFS(JJ)) with a "+" and "-" sign. If a "+" or "-" sign is found in that character location, the target pattern has been found and these two words are tested in Section 3 for proper location of decimal characters, four left of the first sign and five right of the second sign.

If a second sign is not found six characters right of the first, control goes to statement #1000 where J is increased by one and tested to see if it exceeds the number of words in the first BUFS array (i.e., 1000). If so, the last six words in the first BUFS array are placed in the first six words of the second BUFS array. Otherwise, the sign searching loop continues to search for more valid target patterns.

Section 3

To arrive at this section, a target pattern has been found and the task is to analyze the sixteen characters to see if they contain decimal numbers. Assume, for example, that the first word BUFS(1) contain the characters "ABC1234+56" and the second word BUFS(2) contain the characters "789-09876Z". The first sign is the eighth character of BUFS(1); thus N = 48. The computed "GO TO" (following statement #450) directs the test procedure to the appropriate testing statement. For our example, this would send control to statement #490 (N/6 = 48/6 = 8), the eighth statement number of the computed "GO TO".

In statement #490, two words are defined, i.e., "WD1" and "WD2". Following the operations defining WD1, proceed as follows: MASK(N-30) - i.e., MASK(18) yields

111111 111111 111111 000000 000000 000000 000000
000000 000000 000000

.N.MASK(18) gives:

000000 000000 000000 111111 111111 111111 111111
111111 111111 111111

WD, i.e., BUFS(1) yields:

Character	A	B	C	1	2	3	4
Binary NO.	01	02	03	34	35	36	37
Bit	000001	000010	000011	011100	011101	011110	011111

Character	+	5	6
Binary NO.	45	40	41
Bit	100101	100000	100001

WD.A..N.MASK(18) yields:

000000 000000 000000 011100 011101 011110 011111
100101 100000 100001

SHIFT(WD.A..N.MASK(18),N-42) where N-42 = 48-42 = 6, yields:

000000 000000 011100 011101 011110 011111 100101
100000 100001 000000

MASK(N-42), i.e., MASK(6) is:

111111 000000 000000 000000 000000 000000 000000
000000 000000 000000

BUFS(J+1), i.e., BUFS(2) is:

Character	7	8	9	-	0	9	8
Binary NO.	42	43	44	45	33	44	43
Bit	100010	100011	100100	100101	011011	100100	100011
Character	7	6	Z				
Binary NO.	42	41	32				
Bit	100010	100001	011010				

BUFS(J+1).AND.MASK(N-42) yields:

100010 000000 000000 000000 000000 000000 000000
000000 000000 000000

SHIFT(BUFS(J+1).AND.MASK(N-42),N-42) yields:

000000 000000 000000 000000 000000 000000 000000
000000 000000 100010

Finally WD1 = SHIFT(WD.A..N.MASK(N-30),N-42).OR.SHIFT(BUFS(J+1)
.A.MASK(N-42),N-42) yields:

Bit	000000	000000	011100	011101	011110	011111	100101
Binary NO.	00	00	34	35	36	37	45
Character	-	-	1	2	3	4	+
Bit	100000	100001	100010				
Binary NO.	40	41	42				
Character	5	6	7				

Therefore the contents of "WD1" are "--1234+567" (the first twelve bits - two character positions - have no associated characters and are thus interpreted as blanks by the computer).

The next statement after #490 isolates the second thru ninth characters of BUFS(2) (i.e., the "89-09876") in a similar manner and stores them in the last eight character locations of word "WD2". Thus "WD1" contains the counter number, the first sign, and three digits of the X coordinate. "WD2" contains the second sign and the five digits of the Y coordinate. The first two character positions of both words are blanks.

Section 4

In the last section of TRAK 010, the target pattern containing the alleged "valid" data is tested. This operation is implemented in loop #540. The word "TEST" is defined as follows:

"WD1" is shifted left circular (first three character positions then one character at a time) and the logical product taken between each character (in binary) and 77B. This reduces the contents of "TEST" to one character in the last character position. In the word "TEST" for the above example, the binary form is 34B which represents the character "1". When 33B is deducted from 34B the result is 1B. This value is tested to see if it lies between 0 and 13B. Since it does, it is either a sign or a decimal number, thus a valid character. This test continues for each character in the words "WD1" and "WD2". As long as these characters have binary numbers between 33B and 46B (the range of digits and signs), the loop will be completed normally and the contents of words "WD1" and "WD2" will be written on TAPE 7. If one of the binary numbers falls outside this range, it is not a

digit or sign and the test will be aborted. Control will then go to statement #800 and no information will be written on TAPE 7.

Statement #800 commands the current word and the two adjacent words that have failed the sign-digit test to be printed out. This gives the user some information about that data point. If it is a significant point (e.g., peak or valley of an accelerogram trace), he may decide to redigitize that area of the accelerogram or insert a missing digit or sign (if he can be certain of what it should be).

The last part of the TRAK 010 subroutine, checks the value J to see if it has exceeded the size of the BUFS array. When the limit of the BUFS array is finally reached, control returns to statement #10 where another 1000 words are read into memory. The whole process continues until an EOF mark is encountered on the tape. At this point, control returns to the main program PHASE1 where the data stored on TAPE 7 is read into memory.

APPENDIX F

Program Opscan

1.1 Introduction

Program OPSCAN (Optical Scanning) was used to retrieve digitized data placed on 7-track tape by a Perkin-Elmer Microdensitometer. It is listed only for the CDC 6400 computer because the authors have no experience with its use on an IBM computer. The User's Instructions and program which follow were obtained from the Solar Division, Kitt Peak National Observatory. Only the first routine was changed to meet the requirements for accelerogram digitizing.

1.2 User's Instructions

IDENTIFICATION:	Subroutines RUNPID, RUNPSC, RUNSCI, CONVERT REOFS
CONTRIBUTOR:	San Carswell
DATED:	November 5, 1973
PURPOSE:	This set of subroutines is for reading and unpacking data from magnetic tape written by the PDS microdensitometer.
SUBROUTINE:	RUNPID
USAGE:	To read and unpack the ID record at the beginning of each file. CALL RUNPID (NTAPE,IDETC)

Where:

NTAPE is the logical unit number of the mag. tape file holding the PDS data, and

IDETC is an integer array of at least 18 words which will contain the file ID information on return from the subroutine.

The routine will read one record from file NTAPE and if it encounters an End of File, or the record is not the correct length for an ID record, will return the IDETC array as zeros. Otherwise the record is unpacked into the array IDETC in the following way:

IDETC Word(s)		Contents
	1 → 4	= 40 character ID (CDC display code) 4A10
	5	= DX used for scan pattern (microns)
	6	= DY used for scan pattern (microns)
	7	= No of data points per scan
	8	= No of scans requested
	9	= Scan type (CDC display code, E, R or F, A1)
	10	= Speed used for scan (Max=255)
	11	= Position of scan origin (0 = centre, 1,2,3,4. corner)
	12	= Starting corner of scan 1,2,3,4.
	13	= No of mag tape records per scan
	14	= Xtravel (microns) for one scan
	15	= Ytravel (microns) requested
	16	= No of points in each 'full' mag tape record

17 = X co-ordinate of scan origin

18 = Y co-ordinate of scan origin

All values except ID and scan type are returned as integers, ID and scan type are returned in CDC display code.

If any scans in a file are to be unpacked, then RUNPID must be called at the beginning of that file, as the ID is the first record within a file and contains information used by the 'scan unpack' routine.

For all these subroutines the mag. tape reading is done with 'Buffer In' statements, so that the buffer length in the user's 'program' statement for file NTAPE should be the minimum length, i.e. 101 B (octal) or 65 (decimal). If, in any of the subroutines a read (buffer in) encounters a parity error, a message will be printed and the program will terminate with 'STOP 13' printed in the dayfile.

SUBROUTINE: CONVERT If an approximate conversion to intensity is required when a scan is unpacked, then after the call to RUNPID the user should

CALL CONVERT (DNB, GAM)

Where:

DNB is the density of the clear plate and

GAM is the γ of the film or plate.

This call will set a flag so that all calls to RUNPSC until the next End of File is encountered will do the density to intensity conversion. (This means

that CONVERT must be called for each file for which the conversion is required).

SUBROUTINE: RUNPSC To read and unpack one scan line

CALL RUNPSC (IFLIP,SCAN)

Where:

IFLIP is an integer, set = 0 if the user requires that the data values are returned in the order in which they are read, or = 1 if the values are to be returned in reversed order, i.e., 'flipped'. (IFLIP = 1 will normally be used for the even numbered scans of a raster scan pattern,) and

SCAN is a 'real' array of at least NPTS words (where NPTS is the number of points in the scan) and on return from the subroutine will contain the density values or - if CONVERT was called for that file - the intensities. If an End of File is encountered while reading the scan then the first data value will be set = -6400, and no further reading or unpacking is done for that call of the subroutine. The same procedure will be followed if the 'scan start pattern' is not found on the first read.

SUBROUTINE: RUNSCI This is provided as an entry in RUNPSC to enable the user to have integer values returned by the subroutine, i.e.,

CALL·RUNSCI (IFLIP,ISCAN)

Where:

IFLIP is as described for RUNPSC i.e. 0 = no flipping required, 1 = flipping needed and

ISCAN is an integer array of size at least NPTS (where NPTS is the number of points in the scan) which will contain the integer data values on return from the subroutine. (The integer value returned is equal to 400 * density, regardless of whether or not CONVERT was called for that file.)

The 'End of File' encountered, and scan pattern not found procedure is as for RUNSPC, i.e., the first data value is returned as -6400., so that the user should equivalence a real variable to ISCAN (1) and check that variable for negative.

SUBROUTINE: REOFS If the user wishes to ignore files or the remainder of a file it should be done with

CALL REOFS (NTAPE,N)

which will read records on unit NTAPE until N 'End of File' marks have been encountered.

Common block name(s) used

NOOPDS

Other subroutine names within the set UNIN, RUNIN, UNPIN (ENTRY IN UNIN).

STORAGE:	(excluding system routines and buffers) approximately 400 (decimal)

1.3 Program Listing

CONTROL STATEMENTS FOR PROGRAM OPSCAN FOR THE CDC 6400 COMPUTER

```
        MIKLOF,BN        ,CM65000,T80,MT1.
        REQUEST(TAPE1),S,VSN=58578,HY,RO)
        FTN.
        LGO.
7/8/9
```

SOURCE STATEMENTS

```
      PROGRAM OPSCAN(INPUT,OUTPUT,TAPE1,PLOT,TAPE99=PLOT,PUNCH)
      COMMON/NOOPDS/NPTS,NREC,PPERR,NUNIT,MASK,LL,ICON,DN,GAM1,IN(401)
      DIMENSION SCAN(3000),IDETC(18),ISCAN(3000),IACCEL(3,500),NSCAN(500
     .)
      CALL RUNPID(1,IDETC)
      PRINT 200, (IDETC(I),I=1,4)
      PRINT 201, (I,IDETC(I),I=5,8)
      PRINT 202, I,IDETC(9)
      PRINT 201, (I,IDETC(I),I=10,18)
  200 FORMAT (1X,4A10)
  201 FORMAT (1X,2I10)
  202 FORMAT (1X,I10,A10)
      K=0
      NSCANS = IDETC(8)
      DO 224 I=1,200
      CALL RUNPSC(K,SCAN)
      CALL RUNSCI(K,ISCAN)
      M=0
      DO 17 J=1,NPTS
      IF(ISCAN(J).LE.100) 18,17
   18 M=M+1
      NSCAN(M)=J
   17 CONTINUE
       MM=1
      ISUM=0
      J=0
      IF (M .LE. 2) GO TO 54
      DO 3 L=2,M
      IF (IABS(NSCAN(L-1)-NSCAN(L)) .EQ. 1) 2,4
    2 ISUM=ISUM+NSCAN(L-1)
      J=J+1
      IF (L .EQ. M) 5,3
    5 IF (MM-3) 54,55,30
   55 NSCAN(L-1)=NSCAN(L)
    4 IACCEL(MM,I) = (ISUM+NSCAN(L-1))/(J+1)
      MM=MM+1
      ISUM=0
      J=0
    3 CONTINUE
      IF (IACCEL(1,I) .EQ. 0) IACCEL(1,I) = IACCEL(1,I-1)
      IF (IACCEL(2,I) .EQ. 0) IACCEL(2,I) = IACCEL(2,I-1)
      IF (IACCEL(3,I) .EQ. 0) IACCEL(3,I) = IACCEL(3,I-1)
   30 CONTINUE
      K=1
      GO TO 223
  226 K=0
  223 CONTINUE
      IF (IACCEL(1,I) .GT. 1500) IACCEL(1,I) = IACCELL(1,I-1)
      IF (IACCEL(2,I) .LT. 1500 .OR. IACCEL(2,I) .GT. 1650) IACCEL(2,I)
```

```
      = IACCEL(2,I-1)
      IF (IACCEL(3,I) .LT. 1650) IACCEL(3,I) = IACCEL(3,I-1)
      PRINT 204, I, (IACCEL(MM,I), MM=1,3)
  224 CONTINUE
C***** MM IS THE COMPONENT
C***** I IS THE TIME VALUE
C***** IACCEL IS THE ACCELERATION VALUE
      PRINT 204, (I, (IACCEL(MM,I), MM=1,3), I=1,NP)
      PUNCH 40, (I, IACCEL(1,I), I=1,NP)
      PUNCH 40, (I, IACCEL(3,I), I=1,NP)
   40 FORMAT ( 4(2I10))
  200 FORMAT ( 1X,4A10)
  201 FORMAT (1X,2I10)
  202 FORMAT (1X,I10,A10)
  204 FORMAT ( 1X,10I10)
      END
      SUBROUTINE RUNIN(SCAN,N)
      COMMON/NOOPDS/ NPTS,NRECS,PPERR,NUNIT,MASK,LL,ICON,DN,GAMI,IN(401)
      DIMENSION SCAN(5)
C
C           UNPACK LL VALUES AS REAL NOS, DIVIDE THEM BY 400 TO GET THE
C           DENSITY READING, AND DO THE CONVERSION TO INTENSITY IF THE
C           USER HAS REQUESTED IT.
C
C
C
      J=N
      DO 1 I=1,LL
      J=SHIFT(J,12)
      IS=J .AND. MASK
      S=IS/400.
      IF(ICON .EQ. 0) GO TO 1
C
C           IF CONVERSION NOT REQUESTED, RETURN VALUEAS DENSITY

C
C           OTHERWISE, USING DENSITY OF BACKGROUND AND GAMMA OF FILM
C           PREVIOUSLY SUPPLIED BY THE USER, RETURN THE VALUES AS
C           INTENSITIES.
C
      Q=10.**(S-DN)-1.
      S=ABS(Q)
      IF(S .EQ. 0.) GO TO 1
      S=Q* S**GAMI /S
    1 SCAN(I)=S
      RETURN
      END
      SUBROUTINE RUNPID(NTAPE,IDETC)
      DIMENSION IDETC(18),ICDC(27)
C
C
C
C           5TH NOVEMBER 1973
C
C
C           NTAPE IS THE LOGICAL UNIT NO. OF THE FILE FROM WHICH
C           EVERYTHING SHOULD BE READ,  AND IDETC IS AN INTEGER ARRAY
C           OF AT LEAST 18 WORDS THAT WILL CONTAIN THE FOLLOWING INFO.
C           ON EXIT FROM THE SUBROUTINE.
C           ALL VALUES EXCEPT ID AND SCAN TYPE ARE RETURNED AS INTEGERS,
C           ID AND SCAN TYPE ARE RETURNED IN CDC DISPLAY CODE.
C           WORD(S) 1 4   =40 CHARACTER ID (CDC DISPLAY CODE)  4A10
C                   5     =DX USED FOR SCAN PATTERN (MICRONS)
C                   6     =DY USED FOR SCAN PATTERN (MICRONS)
C                   7     =NO OF DATA POINTS PER SCAN
C                   8     =NO OF SCANS REQUESTED.
C                   9     =SCAN TYPE (CDC DISPLAY CODE, E, R OR F,   A1)
C                   10    =SPEED USED FOR SCAN  (MAX=255)
C                   11    =POSITION OF SCAN ORIGIN (O=CENTRE, 1,2,3,4 =
```

```
C                          CORNER)
C                  12   =STARTING CORNER OF SCAN   1,2,3,4.
C                  13   =NO OF MAG TAPE RECORDS PER SCAN
C                  14   =XTRAVEL  (MICRONS) FOR ONE SCAN
C                  15   =YTRAVEL  (MICRONS) REQUESTED
C                  16   =NO OF POINTS IN EACH -FULL- MAG TAPE RECORD
C                  17   =X CO-ORDINATE OF SCAN ORIGIN
C                  18   =Y CO-ORDINATE OF SCAN ORIGIN
C
C
      COMMON/NOOPDS/ NPTS,NRECS,PPERR,NUNIT,MASK,LL,ICON,DN,GAMI,IN(401)
      INTEGER PPERR
      DATA ICDC/62B,64B,60B,53B,55B,67B,61B,51B,52B,47B,45B,56B,46B,57B,
     * 50B,63B,77B,74B,54B,73B,71B,70B,72B,55B,66B,76B,65B/
      DATA MASK/7777B/,ICON/0/
C
C
      NUNIT=NTAPE
      CALL BIN
C
C           READ WOT SHOULD BE THE ID ON THE SPECIFIED UNIT, CHECK THAT
C           THE READ WAS OK AND WHETHER AN EOF WAS READ.
C
      IF(LL .EQ. 0) GO TO 20
C
C           IF YOU HIT AN EOF, THEN JUST RETURN ALL THE ID AS ZERO AND

C
C           HOPE THAT THE USER CHECKS IT
C
   15 DO 16 I=1,18
   16 IDETC(I)=0
      RETURN
   20 IF(LENGTH(NTAPE) .NE. 12) GO TO 15
C
C           THE READ WAS OK, WAS IT THE RIGHT LENGTH FOR AN ID
C           IF NOT, RETURN ID AS ZERO, AS FOR END OF FILE
C
      M=99
      DO 50 I=1,8
      K=IN(I)
      DO 50 J=1,5
      K=SHIFT(K,12)
      L=K .AND. 377B
      N=55B
C
C           UNPACK THE ID PART (IE THE CHARACTERS) AND DUMP THEM IN
C           THE TOP OF THE INPUT ARRAY FOR NOW, FIRST CONVERTING THEM
C           FROM 8 BIT ASCII CODE TO CDC DISPLAY CODE
C
C
C           ANYTHING THAT YOU DONT RECOGNISE, PUT A SPACE CHARACTER IN
      IF(L .LE. 240B .OR. L .GE. 340B) GO TO 49
      N=L-300B
      IF(L .GT. 300B .AND. L .LE. 332B) GO TO 49
      N=L-225B
      IF(L .GE. 260B .AND. L .LE. 271B) GO TO 49
      N=L-240B
      IF(L .GT. 271B) N=L-252B
      IF(L .GT. 332B) N=L-304B
      N=ICDC(N)
   49 M=M+1
   50 IN(M)=N
      M=99
C
C           NOW PACK THE CONVERTED CHARACTERS INTO THE FIRST 4 WORDS OF
C           IDETC, IN A10 FORMAT
C
```

```
      DO 55 I=1,4
      K=0
      DO 53 J=1,10
      M=M+1
   53 K=SHIFT(K,6) .OR. IN(M)
   55 IDETC(I)=K
C
C           NOW UNPACK THE VARIOUS NUMBERS, DX DY ETC
C
      CALL UNIN(IDETC(5),IN(9))
      LL=5
      CALL UNPIN(IDETC(8),IN(10))
      IDETC(9) =SHIFT(IDETC(9)-300B,54) .OR. 55555555555555555B
      DO 60 I=1,2
   60 CALL UNIN(IDETC(10+I*3),IN(10+I))
      DO 65 I=17,18
      IF((IDETC(I) .AND. 40000000B) .NE. 0) IDETC(I)=IDETC(I)-1 .OR.
     * 77777777777700000000B
C
C           ONLY THE X,Y CO/ORDINATES CAN BE NEGATIVE, SO CHECK THEM
C           AND CHANGE FROM TWOS TO ONES COMPLEMENT IF -VE.
   65 CONTINUE
C
C           GET A COPY OF ANY VARIABLES THAT YOU NEED LATER
C
      NPTS=IDETC(7)
      NRECS=IDETC(13)
      PPERR=IDETC(16)
      RETURN
      END
      SUBROUTINE UNIN(ID,N)
      DIMENSION ID(5)
      COMMON/NOOPDS/ NPTS,NRECS,PPERR,NUNIT,MASK,LL,ICON,DN,GAMI,IN(401)
C
C
C           UNPACK THREE THINGS, BITS 1 - 12, BITS 13 - 36, BITS 37 - 60.
C
      ID(1)=SHIFT(N,12) .AND. MASK
      ID(2)=SHIFT(N,-24) .AND. 77777777B
      ID(3)=N .AND. 77777777B
      RETURN
C
      ENTRY UNPIN
C
C           UNPACK LL VARIABLES / DATA POINTS AS INTEGERS, EACH ONE 12
C           BITS LONG.
C
      J=N
      DO 1 I=1,LL
      J=SHIFT(J,12)
    1 ID(I)=J .AND. MASK
      RETURN
      END
      SUBROUTINE CONVERT(DNB,GAM)
      COMMON/NOOPDS/ NPTS,NRECS,PPERR,NUNIT,MASK,LL,ICON,DN,GAMI,IN(401)
C
C           USER MUST SUPPLY DNB = DENSITY OF BACKGROUND, AND GAM = GAMMA
C           OF FILM.
C
C           THIS ROUTINE MUST BE CALLED ONCE FOR EACH FILE FOR WHICH
C           INTENSITIES ARE REQUIRED (INSTEAD OF DENSITIES) AND IT MUST
C           BE CALLED IMMEDIATELY AFTER THE CALL TO RUNPID.
C
      ICON=1
      DN=DNB
      GAMI=1./GAM
      RETURN
      END
```

```
      SUBROUTINE RUNPSC(IFLIP,SCAN)
      COMMON/NOOPDS/ NPTS,NRECS,PPERR,NUNIT,MASK,LL,ICON,DN,GAMI,IN(401)
      DIMENSION SCAN(1)
      INTEGER PPERR
C
C           SCAN IS THE ARRAY TO HOLD THE UNPACKED DENSITIES OR
C           INTENSITIES, AND MUST BE LARGE ENOUGH FOR THE COMPLETE SCAN
C           IFLIP=0 MEANS THAT NO -FLIPPING- IS REQUIRED, IFLIP NON ZERO
C           MEANS THAT THE UNPACKED VALUES SHOULD BE -FLIPPED- BEFORE
C           EXIT FROM THE SUBROUTINE. THIS IS SO THAT RETURN SCANS OF
C           RASTER SCAN PATTERNS MAY BE TURNED AROUND TO LOOK LIKE EDGE
C           SCANS FOR EASE OF USE HEREAFTER.
C
C
      IFINT=0
C
C           SET A FLAG TO SAY THAT USER REQUIRES REAL, UNPACKED VALUES
C           TO BE RETURNED
C
    1 M=1
      LS=-4
      NO=NPTS
C
C           NO = NUMBER OF DATA VALUES LEFT TO UNPACK
C           M= FIRST WORD OF ARRAY -IN- TO UNPACK
C
      DO 180 I=1,NRECS
      CALL BIN
      IF(LL .EQ. 0) GO TO 18
C
   15 ICON=0
C           CLEAR CONVERSION FLAG WHEN EOF FOUND, AND RETURN SCAN(1) AS
C           -6400. SO THAT THE USER KNOWS WOT HAPPENED.
C
C           DO THE SAME THING IF THE CORRECT -SCAN START PATTERN- IS
C           NOT FOUND.
C
      SCAN(1)=-6400.
      RETURN
C
   18 IF(I .NE. 1) GO TO 20
      IF(IN(1) .NE. 77770000777700007777B) GO TO 15
C
C           LOOK FOR THE SCAN START PATTERN ON RECORD 1 OF SCAN.
C
   20 J=NO
      IF(NO .GT. PPERR) J=PPERR
      LL=5
      DO 25 L=1,J,5
      M=M+1
      LS=LS+5
      IF(L+4 .GT. J) LL=J-L+1
      IF(IFINT .NE. 0) GO TO 23
      CALL RUNIN(SCAN(LS),IN(M))
C
C           UNPAK VALUES AS REAL NOS.
C
      GO TO 25
   23 CALL UNPIN(SCAN(LS),IN(M))
C
C           UNPACK THE VALUES AS INTEGERS
C
   25 CONTINUE
      NO=NO-PPERR
  180 M=0
      IF(IFLIP .EQ. 0) RETURN
      IF (IFINT .EQ. 1) RETURN
```

```
C
C          IF THE USER WANTS THEM FLIPPING, DO IT NOW
C
      M=NPTS/2
      L=NPTS
      DO 200 I=1,M
      A=SCAN(I)
      SCAN(I)=SCAN(L)
      SCAN(L)=A
  200 L=L-1
      RETURN
C
C
C          THIS ENTRY FOR BODS WANTING VALUES RETURNED AS INTEGERS.
C
      ENTRY RUNSCI
      IFINT=1
      GO TO 1
C
      END
      SUBROUTINE BIN
C
      COMMON/NOOPDS/ NPTS,NRECS,PPERR,NUNIT,MASK,LL,ICON,DN,GAMI,IN(401)
C
C
C
      BUFFER IN(NUNIT,1)(IN(1),IN(401))
      LL=0
C
C       RETURN LL AS ZERO FOR AN ORDINARY RECORD, AND 1 FOR AN EOF
C
      IF(UNIT(NUNIT)) 22,20,13
C
C          IF YOU GET A PARITY ERROR, SAY SO THEN GIVE UP
C
   14 FORMAT(1H0,20X,*PARITY ERROR ON UNIT*,I4)
   13 PRINT 14,NUNIT
      STOP 13
   20 LL=1
      ICON=0
   22 RETURN
C
      END
      SUBROUTINE REOFS(NTAPE,N)
C
      COMMON/NOOPDS/ NPTS,NRECS,PPERR,NUNIT,MASK,LL,ICON,DN,GAMI,IN(401)
C
C
C
      NUNIT=NTAPE
      DO 10 I=1,N
    5 CALL BIN
      IF(LL .EQ. 0) GO TO 5
C
C          KEEP READING TILL YOU HIT AN EOF
C
   10 CONTINUE
      RETURN
C
      END
```

REFERENCES

1. Halverson, Harry T., "The Strong Motion Accelerograph", Proceedings of the Third World Conference on Earthquake Engineering, New Zealand, 1965, pp. III 75 - III 85.

2. Anderson, J. A., and Wood, H. O., "Description and Theory of the Torsion Seismometer", Bulletin - Seismological Society of America, Vol. 15, March 1925, pp. 1-75.

3. Hudson, Donald E., "Ground Motion Measurements", Earthquake Engineering, Prentice-Hall, Inc., 1970, pp. 112-117.

4. Trifunac, M. D., Brady, A. G., and Hudson, D.E., Strong Motion Earthquake Accelerograms Digitized andPlotted Data, Vol. II, Corrected Accelerograms and Integrated Ground Velocity and Displacement Curves, Part C, Accelerograms IiC041 through IiC055, Earthquake Engineering Research Laboratory, California Institute of Technology, Pasadena, California, February 1973, p. 10.

5. Sixsmith, Edmund C. D., and Roosset, V., "Statistical Properties of Strong Motion Earthquakes", Research Report R70-7, Dept. of Civil Engineering, MIT, Cambridge, Mass., January 1970.

6. Jensclke, V., Clough, R. W., and Pensier, J., "Analysis of Earth Motion Accelerograms", Structural and Materials Research Report SES M64-1, Dept. of Civil Engineering, University of California, Berkeley, California, January 1964.

7. Boyce, W. H., "Analysis of Earthquake Accelerograms", Bulletin No. 1, Civil Engineering Department, PAPUA and New Guinea Institute of Technology, New Guinea, 1970.

8. Brady, A. G., Hudson, D. E., Trifunac, M. D., Strong Motion Earthquake Accelerograms, Digitized and Plotted Data, Vol. I, Part C, Accelerograms from the San Fernando, California Earthquake, Report No. EERL 71-20, California Institute of Technology, Earthquake Engineering Research Laboratory, Pasadena, California, July 1971.

9. Trifunac, M. D., Low Frequency Digitization Errors and a New Method for Zero Baseline Correction of Strong Motion Accelerograms, Report No. EERL 20-07, California Institute of Technology, Earthquake Engineering Research Laboratory, Pasadena, California, September, 1970.

10. Trifunac, M. D., and Lee, V., Routine Computer Processsing of Strong Motion Accelerograms, Report No. EERL 73-03, California Institute of Technology, Earthquake Engineering Research Laboratory, October, 1973. Available as NTIS Report PB-226047.

11. Trifunac, M. D., Udwadia, F.E., and Brady, A.G., High Frequency Error and Instrument Corrections of Strong-Motion Accelerograms, Report No. EERL 71-05, California Institute of Technology, Earthquake Engineering Research Laboratory, July, 1971.

12. Trifunac, M.D., Brady, A.G., and Hudson, D.E., Strong Motion Earthquake Accelerogram Digitized and Method Data,Vol. II., "Corrected Accelerogram and Integrated Ground velocity and Displacement Curves, Part A, Accelerograms II A001 through 4A020," Earthquake Engineering Research Laboratory, Cal. Tech., Pasadena, California, September, 1971, p. 8-25.

13. Trifunac, M.D., et. al., "Analysis of Errors in Digitized Strong Motion Accelerograms," Bulletin of the Seismological Society of America, February, 1973, pp 157-187.

14. Ormsby, J.F., "Design of Numerical Filters with Application to Missile Data Processing," Journal of the Association for Computer Machinery, 1961, pp. 440-447.

15. Halloway, L.I., "Smooting and Filtering of Time Series and Space Fields," Advances in Biophysics, No. 4, 1958, pp. 351-389.

16. Reference 12, p. 17.

17. Reference 12, p. 18.

18. Berg, G.V., and Housner, G.W., "Integrated Velocity and Displacement of Strong Earthquake Ground Motion", Bulletin of the Seismological Society of America, Vol. 37, April, 1961, pp. 175-189.

19. Reference 3, p. 111.

20. Trifunac, M.D., and Lee, V.W., "Note on the Accuracy of Computed Ground Displacements from Strong-Motion Accelerograms," Bulletin of the Seismological Society of America, Vol. 64, No. 4, August, 1974, pp. 1209-1219.

21. Protter, M.H., and Morrey, C.B., Jr., College Calculus with Analytic Geometry, Second Edition, Addison Wesley Publishing Co., Reading, Mass. 1970, p. 546.

*U.S. GOVERNMENT PRINTING OFFICE : 1977 0-731-044/11

FHWA
R&D